ITS
Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR - 141501

PENGEMBANGAN PENCARIAN PRODUK TERKAIT MENGGUNAKAN EUCLIDEAN DISTANCE DAN COSINE SIMILARITY PADA APLIKASI HALAL NUTRITION FOOD

DEVELOPMENT OF RELATED PRODUCT FEATURE FOR HALAL NUTRITION FOOD APPLICATION USING EUCLIDEAN DISTANCE AND COSINE SIMILARITY

AZMI ADI FIRMANSYAH
NRP 5214100045

Dosen Pembimbing
Nur Aini Rakhmawati, Ph.D

DEPARTEMEN SISTEM INFORMASI
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya, 2018

Halaman ini sengaja dikosongkan

TUGAS AKHIR - 141501

PENGEMBANGAN PENCARIAN PRODUK TERKAIT MENGGUNAKAN EUCLIDEAN DISTANCE DAN COSINE SIMILARITY PADA APLIKASI HALAL NUTRITION FOOD

AZMI ADI FIRMANSYAH
NRP 5214100045

Dosen Pembimbing
Nur Aini Rakhmawati, Ph.D

DEPARTEMEN SISTEM INFORMASI
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya, 2018

Halaman ini sengaja dikosongkan

UNDERGRADUATE THESIS - 141501

DEVELOPMENT OF RELATED PRODUCT FEATURE FOR HALAL NUTRITION FOOD APPLICATION USING EUCLIDEAN DISTANCE AND COSINE SIMILARITY

AZMI ADI FIRMANSYAH
NRP 5214100045

Supervisor
Nur Aini Rakhmawati, Ph.D

DEPARTMENT OF INFORMATION SYSTEM
Faculty of Information Communication and Technology
Institut Teknologi Sepuluh Nopember
Surabaya, 2018

Halaman ini sengaja dikosongkan

LEMBAR PENGESAHAN

PENGEMBANGAN PENCARIAN PRODUK TERKAIT MENGGUNAKAN EUCLIDEAN DISTANCE DAN COSINE SIMILARITY PADA APLIKASI HALAL NUTRITION FOOD

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Akuisisi Data dan Diseminasi Informasi
Program Studi S1 Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh :

AZMI ADI FIRMANSYAH
NRP: 5214100045

Surabaya, Januari 2018

Plh Kepala
DEPARTEMEN SISTEM INFORMASI

Edwin Riksakomara, S.Kom., M.T
NIP. 196907252003121001

Halaman ini sengaja dikosongkan

LEMBAR PERSETUJUAN

PENGEMBANGAN PENCARIAN PRODUK TERKAIT MENGGUNAKAN EUCLIDEAN DISTANCE DAN COSINE SIMILARITY PADA APLIKASI HALAL NUTRITION FOOD

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Akuisisi Data dan Diseminasi Informasi
Program Studi S1 Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh :

AZMI ADI FIRMANSYAH
NRP: 05211440000045

Disetujui Tim Penguji: Tanggal Ujian: 12 Januari 2018
Periode Wisuda: Maret 2018

Nur Aini Rakhmawati, Ph.D **(Pembimbing 1)**

Faizal Johan Atletiko, S.Kom, M.T **(Penguji 1)**

Radityo Prasetianto.W, S.Kom, M.Kom **(Penguji 2)**

Halaman ini sengaja dikosongkan

PENGEMBANGAN PENCARIAN PRODUK TERKAIT MENGGUNAKAN EUCLIDEAN DISTANCE DAN COSINE SIMILARITY PADA APLIKASI HALAL NUTRITION FOOD

Nama	: AZMI ADI FIRMANSYAH
NRP	: 5214100045
Departemen	: Sistem Informasi FTIK
Pembimbing I	: Nur Aini Rakhmawati, Ph.D

Abstrak

Seiring berjalannya waktu muncul lebih banyak inovasi dan variasi produk makanan baru. Namun hanya sebagian kecil yang sudah tersertifikasi. Pada tahun 2016 lalu, telah dikembangkan aplikasi Halal Nutrition Food untuk oleh Jauhar Fatawi. Aplikasi tersebut mempermudah pengguna melakukan pencarian terhadap produk halal. Aplikasi dikembangkan lagi pada tahun 2017 oleh Adnan Mauludin Fajriyadi dengan mengembangkan fitur pencarian menggunakan algoritma OKAPI BM25F untuk meningkatkan relevansi hasil pencarian.

Penelitian kali ini mengembangkan penelitian sebelumnya dengan menampilkan produk halal yang terkait berdasarkan komposisinya. Produk terkait dicari berdasarkan kemiripan komposisi menggunakan euclidean distance dan cosine similarity. Produk yang memiliki banyak kemiripan dengan produk yang telah tersertifikasi, dapat menambah keyakinan pengguna walaupun produk yang sedang dicari belum tersertifikasi.

Aplikasi dapat menampilkan notifikasi kepada pengguna apabila terdapat produk halal yang mirip dengan produk yang sedang dilihat pengguna. Penelitian ini juga menunjukkan bahwa pencarian

produk terkait menggunakan cosine similarity memiliki presisi sebesar 84%, sedangkan euclidean distance memiliki presisi sebesar 72%. Penelitian ini juga menguji fitur MoreLikeThis dari Apache Lucene yang memiliki presisi sebesar 80%, sedikit lebih rendah dibandingkan dengan cosine similarity.

Kata kunci: linked data, produk terkait, halal, cosine similarity, euclidean distance, morelikethis.

DEVELOPMENT OF RELATED PRODUCT FEATURE FOR HALAL NUTRITION FOOD APPLICATION USING EUCLIDEAN DISTANCE AND COSINE SIMILARITY

Name	: AZMI ADI FIRMANSYAH
NRP	: 5214100045
Major	: Information System FTIK
Supervisor I	: Nur Aini Rakhmawati, Ph.D

Abstract

As time went on, more innovations and new food product variations emerged. But only a small percentage has been certified. In 2016, Halal Nutrition Food has been developed by Jauhar Fatawi. The app makes it easy for users to search halal products. The application was developed again in 2017 by Adnan Mauludin Fajriyadi by developing a search feature using the BM25F OKAPI algorithm to improve the relevance of search results.

This research develops previous research by showing the related halal product based on its composition. Related products are searched on the basis of similarities of composition using euclidean distance and cosine similarity. Products that have many similarities with a certified product can add to the user's belief that the product being searched is not certified.

The app can display a notification to the user if there is a halal product similar to the product that the user is viewing. This study also shows that the search for related products using cosine similarity has a precision of 84%, while the euclidean distance has a precision of 72%. This study also tested MoreLikeThis feature from Apache Lucene that has a precision of 80%, slightly lower than

cosine similarity

Keywords: linked data, similar product, halal, cosine similarity, euclidean distance, morelikethis

KATA PENGANTAR

Segala puji dan syukur pada Allah SWT yang telah melimpahkan rahmat dan anugerah-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul "Pengembangan Pencarian Produk Terkait menggunakan Euclidean Distance dan Cosine Similarity pada Aplikasi Halal Food Nutrition" dengan tepat waktu.

Harapan dari penulis semoga apa yang tertulis di dalam buku Tugas Akhir ini dapat bermanfaat bagi pengembangan ilmu pengetahuan saat ini, serta dapat memberikan kontribusi nyata bagi kampus Sistem Informasi, ITS, dan bangsa Indonesia.

Dalam pelaksanaan dan pembuatan Tugas Akhir ini tentunya sangat banyak bantuan yang penulis terima dari berbagai pihak, tanpa mengurangi rasa hormat penulis ingin menyampaikan terimakasih kepada:

1. Ibu Nur Aini Rakhmawati, S.Kom., M.Sc., Eng. selaku dosen pembimbing penulis yang telah memberikan ide, bimbingan, saran, kritik, ilmu, dan pengalamannya yang sangat bermanfaat sehingga penulis dapat menyelesaikan Tugas Akhir ini.
2. Bapak Bekti Cahyo Hidayanto, S.Si, M.Kom selaku dosen wali penulis yang selalu membimbing dan memberikan arahan ke penulis.
3. Bapak Dr. Ir. Aris Tjahyanto, M.Kom. selaku Kepala Departemen Sistem Informasi yang telah memberikan ilmu dan pengalaman kepada penulis.
4. Seluruh dosen Departemen Sistem Informasi ITS yang telah memberikan ilmu pengetahuan dan pengalaman yang sangat berharga dan bermanfaat bagi penulis.
5. Seluruh keluarga besar saya khususnya (Alm) Bapak, Ibu dan Kakak saya yang telah memberikan dukungan baik material

maupun non material serta semangat kepada penulis hingga akhirnya dapat menyelesaikan tugas akhir ini.

6. Teman-teman Sistem Informasi angkatan 2014 (OSIRIS) yang senantiasa menemani dan memberikan motivasi bagi penulis selama perkuliahan hingga dapat menyelesaikan tugas akhir, khususnya teman-teman UKM Cinta Lab, The Brilliant Class, Jamaah LPSI, dan teman-teman lain yang tidak bisa penulis sebutkan satu per satu.
7. Kakak dan adik angkatan 2012, 2013, 2015, 2016 dan 2017 yang selalu membantu dan memberikan semangat bagi penulis.
8. Serta seluruh pihak-pihak lain yang tidak dapat disebutkan satu per satu yang telah banyak membantu penulis selama perkuliahan hingga dapat menyelesaikan tugas akhir ini.

Tugas Akhir ini merupakan persembahan bagi penulis untuk kedua orang tua dan keluarga besar yang selalu memberikan motivasi terbaik bagi penulis untuk dapat menuntut ilmu setinggi-tingginya dan dapat meraih kesuksesan.

Tugas Akhir ini juga masih jauh dari kata sempurna, sehingga penulis mengharapkan saran dan kritik yang membangun dari pembaca untuk perbaikan ke depan. Semoga Tugas Akhir ini dapat bermanfaat bagi perkembangan ilmu pengetahuan dan semua pihak.

DAFTAR ISI

Halaman ini sengaja dikosongkan

DAFTAR TABEL

Halaman ini sengaja dikosongkan

DAFTAR GAMBAR

Halaman ini sengaja dikosongkan

BAB 1

PENDAHULUAN

Pada bab pedahluan ini akan membahas terkait latar belakang masalah, perumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan relevansi terhadap pengerjaan tugas akhir.

1.1 Latar Belakang

Pada tahun 2010 tercatat sebanyak 87,18 persen penduduk Indonesia beragama Islam, [12]. Bagi masyarakat muslim, terdapat ketentuan syariat yang mengatur segala urusan kehidupan, termasuk dalam hal makanan. Masyarakat muslim diwajibkan untuk makan makanan yang halal dan baik. Di Indonesia, penentuan halal haram suatu makanan dibantu oleh organisasi Lembaga Pengkajian Pangan Obat-obatan dan Kosmetika Majelis Ulama Indonesia (LPPOM MUI).

LPPOM MUI merupakan lembaga yang dibentuk oleh Majelis Ulama Indonesia yang mempunyai fungsi utama melaksanakan sertifikasi halal. Untuk mendapatkan sertifikat halal, industri pengolahan (pangan, obat, kosmetika), Rumah Potong Hewan (RPH), dan restoran/katering/dapur harus mendaftar ke website LPPOM MUI terlebih dahulu dan mengikuti prosedur yang tersedia. Hingga tahun 2015, LPPOM MUI telah mencatat sebanyak 309115 produk yang tersertifikasi halal [9]. Informasi mengenai produk halal dapat diakses pada situs http://halalmui.org.

Seiring berjalannya waktu muncul lebih banyak inovasi dan variasi produk baru. Namun hanya sebagian kecil yang sudah tersertifikasi

[10]. Meskipun belum tersertifikasi, produk tersebut tidak bisa serta merta dikatakan tidak halal.

Pada tahun 2016 lalu, telah dikembangkan aplikasi Halal Nutrition Food oleh Jauhar Fatawi. Beliau melakukan penelitian mengenai produk halal dan membangun aplikasi Halal Nutrition Food yang dapat mempermudah pengguna melakukan pencarian terhadap produk halal. Aplikasi tersebut kemudian dikembangkan lagi pada tahun 2017 oleh Adnan Mauludin Fajriyadi. Beliau mengembangkan fitur pencarian menggunakan algoritma OKAPI BM25F untuk meningkatkan relevansi hasil pencarian.

Penelitian ini mengembangkan dari dua penelitian sebelumnya. Penelitian ini mencoba untuk menganalisa tingkat kemiripan suatu produk yang belum tersertifikasi dengan produk yang telah tersertifikasi segi komposisinya. Komposisi produk yang dicari akan dibandingkan dengan komposisi dari tiap produk yang ada dalam database. Untuk membandingkan produk tersebut, digunakan euclidean distance dan cosine similarity antara dua produk. Sebelumnya, tiap produk akan dihitung bobotnya menggunakan tf-idf untuk mempermudah proses perbandingannya. Bobot kemudian dihitung menggunakan euclidean distance dan cosine similarity untuk mengetahui kemiripan dari dua produk. Nilai euclidean distance dan cosine similarity kemudian diranking dan diambil 5 produk teratas dari masing-masing algoritma.

Dari hasil perbandingan tersebut akan diketahui seberapa mirip suatu produk terhadap produk yang telah tersertifikasi halal. Produk terkait akan ditampilkan pada saat pengguna melihat detil produk yang sedang dicari. Produk yang memiliki banyak kemiripan dengan produk yang telah tersertifikasi, dapat menambah keyakinan pengguna, walaupun produk yang sedang dicari belum tersertifikasi

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, berikut adalah rumusan masalah dalam tugas akhir ini :

1. Bagaimana pengguna dapat mengetahui produk yang terkait dengan produk yang sedang dicari.
2. Bagaimana menerapkan algoritma euclidean distance dan cosine similarity dalam menentukan produk yang terkait dengan produk yang sedang dicari di aplikasi Halal Nutrition Food.

1.3 Batasan Masalah

Tugas akhir ini memiliki terbatas pada hal-hal sebagai berikut :

1. Hasil akhir dari tugas akhir ini yaitu penambahan fitur pada aplikasi Halal Nutrition Food untuk menampilkan produk yang terkait dengan produk yang sedang dicari.
2. Perhitungan keterkaitan pada produk dihitung menggunakan pembobotan TF-IDF dan menggunakan perbandingan *euclidean distance* dan *cosine similarity*.
3. Perhitungan keterkaitan tidak dilakukan secara live query, namun dihitung setiap bulan.

1.4 Tujuan

Tujuan dari penelitian ini adalah untuk mengembangkan aplikasi Halal Nutrition Food dengan menambahkan fitur untuk menampilkan produk yang terkait dengan produk yang dicari oleh pengguna.

1.5 Manfaat

Manfaat yang diharapkan dari penelitian ini adalah :

1. Bagi pengguna, dapat memberikan rekomendasi mengenai produk yang terkait dengan produk yang sedang dicari berdasarkan keterkaitan dari komposisi produk tersebut.
2. Bagi peneliti, dapat memberikan sumbangsih terhadap penelitian linked data yang dapat lebih dikembangkan lagi nantinya.

1.6 Relevansi

Tugas akhir ini berkaitan dengan pengembangan perangkat lunak dengan metode penyimpanan dan publikasi data berupa Linked Open Data ditambah dengan pencarian produk terkait menggunakan cosine similarity. Tugas akhir ini berkaitan dengan mata kuliah, Pemrograman Berbasis Web, Penggalian Data dan Analitika Bisnis, dan Pemrograman Integratif.

Tugas akhir ini layak dijadikan sebagai tugas akhir pada tingkat S1, karena tugas ini memecahkan masalah yaitu dalam hal membantu menemukan produk yang terkait dalam aplikasi Halal Nutrition Food.

BAB 2

TINJAUAN PUSTAKA

Pada bab ini dijelaskan mengenai teori-teori terkait yang bersumber dari buku, jurnal, ataupun artikel yang berfungsi sebagai dasar dalam melakuk an pengerjaan tugas akhir agar dapat memahami konsep atau teori penyelesaian permasalahan yang ada.

2.1 Penelitian Sebelumnya

Penelitian "Text Classification and Named Entities for New Event Detection" oleh Giridhar Kumaran dan James Allan [7]. Peneliti mencoba melakukan klasifikasi teks dengan *New Event Detection (NED)* dan juga menggunakan *named entity*. Klasifikasi dilakukan dengan mengukur tingkat cerita yang tumpang tindih antara cerita baru dengan cerita yang pernah ada di masa lalu. Dalam menetukan kesamaan antara cerita lama dan cerita yang akan diidentifikasi, peneliti menggunakan metrik *cosine similarity*. Sebelum dilakukan perhitungan *cosine similarity*, dilakukan *preprocessing* data terlebih dahulu. Kemudian isi cerita di-*tokenize* terlebih dahulu, lalu menghapus *stop words*, *stemming* menggunakan algoritma *K-stem* dan membuat *document vectors*. Pembobotan nilai dilakukan menggunakan *TF-IDF* dan dilakukan *document similarity normalization* terlebih dahulu sebelum nilai akhir diberikan pada sebuah cerita. Peneliti mencoba meningkatkan hasil dengan mengelompokkan cerita terlebih dahulu menggunakan *Linguistic Data Consortium (LDC)* dan juga membedakan pembobotan pada *named entity*. Hasil penelitian menunjukkan bahwa pengelompokan menggunakan *LDC* dan pembobotan yang berbeda pada *named entity* dapat menurunkan tingkat kesalahan deteksi.

Penelitian ”Klasifikasi Berita Online dengan menggunakan Pembobotan TF-IDF dan Cosine Similarity” oleh Bening Herwijayanti, Dian Eka Ratnawati, dan Lailil Muflikhah [5]. Peneliti mencoba melakukan klasifikasi berita online yang didapat dari situs *kompas.com* menggunakan pembobotan tf-idf dan *cosine similarity*. Peneliti mencoba melakukan klasifikasi karena terkadang berita tidak dipublikasikan sesuai dengan kategorinya. Klasifikasi juga dianggap penting bagi user untuk mencari berita sesuai dengan kategorinya. Klasifikasi ini menggunakan teknik *text mining*, yaitu analisis data berupa teks yang bersumber dari dokumen. Peneliti mengumpulkan data terlebih dahulu kemudian melakukan pembersihan data mulai dari *tokenizing, stop words removal* dan *stemming*. Setelah data dibersihkan selanjutnya data di-transform yaitu diberi pembobotan menggunakan tf-idf dilanjutkan dengan reduksi data. Peneliti melakukan 4 kali pengujian menggunakan 46 data. Tiap pengujian menggunakan jumlah data latih dan data uji yang berbeda-beda. Dari keempat hasil pengujian diketahui bahwa tingkat akurasi bertambah seiring dengan bertambahnya jumlah data uji. Rata-rata dari 4 pengujian tersebut mendapatkan akurasi sebesar 91.25 persen.

Penelititan ”Rancang Bangun Sistem Pencarian dalam Aplikasi Halal Nutrition Food Menggunakan Algoritma OKAPI BM25F” oleh Adnan Mauludin Fajriyadi [3]. Penelitian ini merupakan penelitian yang dilanjutkan dalam tugas akhir ini. Dalam penelitian ini, peneliti menggunakan algoritma OKAPI BM25F guna meningkatkan relevansi hasil pencarian terhadap data produk halal yang terdapat pada aplikasi Halal Nutrition Food. Algoritma OKAPI BM25F akan memberikan ranking untuk setiap term yang terdapat dalam dokumen dan menampilkan hasil query urut berdasarkan ranking term tersebut.

Untuk lebih jelasnya terkait penelitian sebelumnya yang telah dilakukan dapat dilihat pada tabel 2.1.

Tabel 2.1: Perbandingan studi sebelumnya.

Nama Peneliti	Judul Penelitian	Isi Penelitian	Keterkaitan dengan Tugas Akhir
Giridhar Kumaran dan James Allan	Text Classification and Named Entities for New Event Detection	Peneliti mencoba melakukan klasifikasi teks dengan New Event Detection (NED) dan juga menggunakan *named entities* produk. Peneliti menggunakan pembobotan tf-idf dan perbandingan cosine similarity [7]	Sama halnya dalam penggunaan tf-idf dan cosine similarity
Bening Herwijayanti, Dian Eka Ratnawati, dan Lailil Muflikhah3	Klasifikasi Berita Online dengan menggunakan Pembobotan TF-IDF dan Cosine Similarity	Peneliti mencoba mengklasifikasikan berita pada situs online menggunakan tf-idf dan cosine similarity [5]	Kesamaan dalam penggunaan tf-idf dan cosine similarity.
Adnan Mauludin Fajriyadi	Rancang Bangun Sistem Pencarian dalam Aplikasi Halal Nutrition Food Menggunakan Algoritma OKAPI BM25F	Peneliti mengembangkan lagi aplikasi Halal Nutrition Food dengan meningkatkan relevansi pencarian menggunakan algoritma OKAPI BM25F [3]	Penelitian yang dilanjutkan dalam tugas akhir dengan menambahkan fitur menampilkan produk terkait

2.2 Dasar Teori

Berikut di bawah ini adalah teori-teori dan istilah yang ada dalam penelitian ini

2.2.1 Halal Nutrition Food

Aplikasi Halal Nutrition Food merupakan aplikasi yang dikembangkan pada tahun 2016 oleh Jauhar Fatawi. Beliau membangun aplikasi tersebut sebagai hasil dari penelitiannya yang berjudul ”Rancang Bangun Perangkat Lunak Linked Open Data Halal dan Gizi pada Produk Makanan dan Minuman”. Aplikasi Halal Nutrition Food dapat mempermudah pengguna melakukan pencarian terhadap produk halal. Perbedaan fitur pencarian pada aplikasi Halal Nutrition Food dengan pada situs http://halalmui.org adalah adanya informasi tambahan berupa komposisi penyusun dan zat aditif yang terdapat pada produk halal. Informasi tentang komposisi dan zat aditif dihubungkan dengan dataset PubChem, DBpedia, dan MeSH. Dataset produk halal disimpan dalam format RDF. Saat pengguna melakukan pencarian, aplikasi akan melakukan query SPARQL ke dalam database untuk menampilkan produk yang sedang dicari [4].

Aplikasi ini kemudian dikembangkan lagi oleh Adnan Mauludin Fajriyadi. Beliau mengembangkan aplikasi Halal Nutrition Food juga sebagai hasil dari penelitian yang berjudul ”Rancang Bangun Sistem Pencarian dalam Aplikasi Halal Nutrition Food menggunakan algoritma OKAPI BM25F”. Penelitian tersebut mencoba mengoptimalkan fungsi pencarian dengan meranking dokumen menggunakan algoritma OKAPI BM25F sehingga hasil pencarian yang didapat lebih relevan [3].

2.2.2 Linked data

Istilah Linked Data mengacu pada serangkaian cara terbaik untuk menampilkan data terstruktur di Web. Prinsip-prinsip ini telah diciptakan oleh Tim Berners-Lee dalam catatan desain. Prinsipnya adalah:

1. Menggunakan URI sebagai nama untuk berbagai hal
2. Menggunakan URI HTTP sehingga orang bisa mencari nama tersebut.
3. Berikan informasi yang berguna lewat URI dengan menggunakan standar.
4. Sertakan tautan ke URI lainnya sehingga pengguna dapat menemukan lebih banyak hal melalui URI tersebut.

Gagasan awal dari prinsip-prinsip ini adalah agar menggunakan standar representasi dan akses terhadap data yang ada pada Web. Di sisi lain, mengatur hyperlink antara data dari sumber yang berbeda. Hyperlink menghubungkan semua Linked Data ke dalam satu grafik data global. Dengan demikian, LinkedData mengubah dokumen hyertext menjadi spreadsheet dan database[1].

2.2.3 Tf-idf

Tf-idf merupakan singkatan dari term frequency-inverse document frequency. Term frequency, dinyatakan dengan $tf_{t,d}$ menunjukkan jumlah term t yang muncul pada dokumen d. Document frequency, dinyatakan dengan df_t menunjukkan dokumen di dalam koleksi yang memiliki term t. Sedangkan Inverse document frequency, dinyatakan dengan idf_t memiliki formula sebagai berikut :

$$idf_t = \log\left(\ \frac{N}{df_t}\ \right) \tag{2.1}$$

Dimana :
N : jumlah seluruh dokumen (koleksi)
df_t : jumlah dokumen dengan term t dari seluruh dokumen

Dengan menggunakan inverse document frequency, maka nilai term yang sering muncul memiliki nilai idf rendah, dan term yang jarang muncul memiliki nilai idf tinggi[8]. Sehingga formula pembobotan gabungan tf-idf dapat dituliskan sebagai berikut :

$$w_{t,d} = tf_{t,d} \times \log\left(\ \frac{N}{df_t}\ \right) \tag{2.2}$$

Dimana :
$w_{t,d}$: bobot untuk term t pada dokumen d
$tf_{t,d}$: jumlah kemunculan term t pada dokumen d
N : jumlah seluruh dokumen (koleksi)
df_t : jumlah dokumen dengan term t dari seluruh dokumen

Formula di atas dapat diartikan bahwa sebuah term akan memiliki bobot yang tinggi pada dokumen yang memiliki banyak kemunculan term tersebut, dan term tersebut muncul hanya pada sedikit dokumen.

Delbru menyajikan adaptasi pembobotan untuk linked data dalam bentuk *term frequency - inverse entity frequency* atau TF-IEF [2]. Pembobotan ini mengukur pentingnya sebuah term di seluruh entity dalam koleksi, yang dijelaskan dalam persamaan sebagai berikut :

$$tf - ief(t,e) = \frac{n_{t,e}}{\sum_k n_{k,e}} \times \log \frac{\mid V^E \mid}{1 + n_{t,e}} \tag{2.3}$$

dengan $n_{t,e}$ merupakan jumlah kemunculan term *t* dalam entity *e*, $\sum_k n_{k,e}$ menyatakan jumlah term dalam dokumen (panjang dokumen), dan V^E merupakan jumlah entitas di dalam koleksi.

2.2.4 Euclidean distance

Euclidean distance merupakan salah satu motode dengan pendekatan geometris. Euclidean distance menghitung jarak terdekat dari dua titik secara langsung. Formula euclidean distance dapat dituliskan sebagai berikut :

$$dist(p,q) = \sqrt{\sum_{1}^{M}(p-q)^2} \tag{2.4}$$

Euclidean distance juga dapat dinyatakan dalam metrik *Minkowski* L_p *distance* [13] :

$$dist(p,q) = (\sum_{k=1}^{n} \mid p_k - q_k \mid^2)^{\frac{1}{2}} \tag{2.5}$$

$$sim(p,q) = \frac{1}{dist(p,q)+1} \tag{2.6}$$

Dengan p dan q masing-masing adalah dua titik yang akan dicari jaraknya

Titik p dan q disubstitusikan dengan bobot tf-idf yang telah dihitung untuk term t pada dokumen d, sehingga formulanya menjadi :

$$dist(d,d') = \sqrt{\sum_{t=1}^{n}(weight(d_t) - weight(d'_t))^2} \tag{2.7}$$

Dimana :
$weight(d_t)$: bobot tf-idf pada term t di dokumen d

Similarity pada euclidean distance memiliki rentang nilai antara 0 hingga 1. Angka 0 menyatakan bahwa dua produk tidak memiliki kemiripan dan angka 1 menyatakan bahwa dua produk tersebut identik.

2.2.5 Cosine similarity

Cosine similarity merupakan salah satu contoh perhitungan kemiripan dengan pendekatan geometris[13]. Dalam pendekatan geometris, sebuah dokumen dianggap sebagai titik-titik dalam ruang multi dimensi. Dua dokumen kemudian dibandingkan berdasarkan lokasinya di ruang multi dimensi. Dalam ruang multi dimensi, sebuah term memiliki satu sumbu [8]. Banyaknya term dalam suatu dokumen mewakili komponen vektor.

Perhitungan kemiripan antara dua dokumen dilakukan dengan menghitung kemiripan nilai cosinus dari representasi vektor dua dokumen. Representasi vektor digunakan untuk mempermudah perhitungan terhadap dokumen yang panjang. Formula cosine similarity dapat dituliskan sebagai berikut :

$$Sim(d, d') = \frac{\overrightarrow{V}(d_1) \cdot \overrightarrow{V}(d_2)}{\mid \overrightarrow{V}(d_1) \mid\mid \overrightarrow{V}(d_2) \mid'} \tag{2.8}$$

Dimana bilangan pembilang merupakan hasil perkalian $dot\ product$ antara dokumen d_1 dengan dokumen d_2, dan penyebut merupakan hasil kali nilai $Euclidian\ distance$ (jarak antara dokumen d_1 dan d_2)

Hasil perkalian $dot\ product$ dapat dituliskan menjadi

$$\sum_{1}^{M} \overrightarrow{V}(d_1)\overrightarrow{V}(d_2) \tag{2.9}$$

dan nilai *Euclidian distance* untuk vektor *d* dapat dituliskan menjadi

$$\sqrt{\sum_{1}^{M} \overrightarrow{V}^2(d)} \tag{2.10}$$

Komponen vektor merupakan hasil pembobotan yang telah dihitung menggunakan tf-ief, sehingga formula $cosine\ similarity$ dapat dituliskan menjadi :

$$Sim(d, d') = \frac{\sum_w weight(w, d) * weight(w, d')}{\sqrt{\sum_w weight(w, d)^2}\sqrt{\sum_w weight(w, d')^2}} \tag{2.11}$$

Dimana :
$weight(w, d)$: bobot dengan nilai tf-ief w pada dokumen d

Cosine similarity memiliki rentang nilai antara 0 hingga 1. Angka 0 menyatakan bahwa dua produk tidak memiliki kemiripan dan angka 1 menyatakan bahwa dua produk bersifat identik.

2.2.6 Apache Lucene

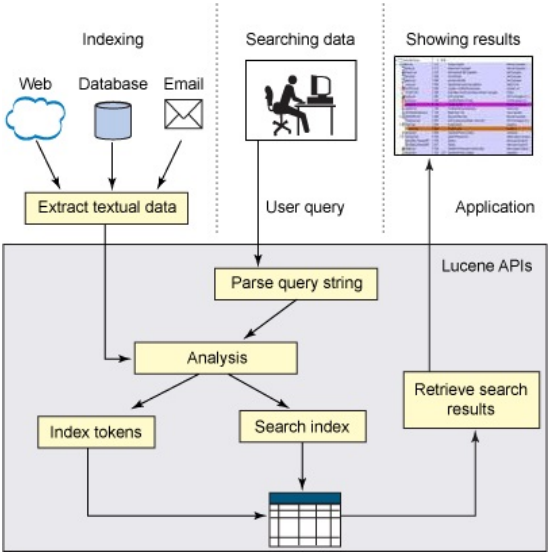
Apache Lucene merupakan library mesin pencari teks dengan fitur lengkap dan performa tinggi yang dituliskan menggunkan bahasa Java. Teknologi ini cocok untuk semua aplikasi yang membutuhkan fitur *full-text search*, terutama untuk aplikasi cross platform.

Hal tersebut dilakukan dengan menambahkan konten dari dokumen ke dalam indeks *full-text*. Hal Ini untuk dilakukan kueri pada indeks, dan mengembalikan hasil berdasarkan relevansi dengan kueri atau diurutkan menurut atribut lainnya seperti tanggal modifikasi terakhir pada dokumen. Konten yang ditambahkan ke Lucene bisa

berasal dari berbagai sumber, seperti database SQL / NoSQL, file sistem, atau bahkan dari situs web.

Lucene mampu mendapatkan hasil pencarian yang cepat karena melakukan pencarian terhadap indeks daripada mencari teks secara langsung. Indeks ini disebut indeks terbalik (*inverted index*), karena membalikkan struktur data yang berpusat pada halaman menjadi struktur data yang berpusat pada kata kunci[6].

Pada gambar 2.1 dijelaskan proses yang ada di dalam Apache Lucene [11]:



Gambar 2.1: Proses dalam Apache Lucene

Berikut adalah fitur dari Lucene :

1. Perankingan terhadap hasil pencarian
2. Mendukung banyak tipe kueri

3. Pencarian pada field dokumen (judul, author, konten)
4. Pengurutan berdasarkan field
5. Pencarian multi-indeks
6. Proses pencarian dan update secara simultan
7. Mesin penyimpanan yang dapat dikonfigurasi
8. Memiliki model ranking bawaan, meliputi Vector Space Model and Okapi BM25

2.2.7 MoreLikeThis

MoreLikeThis merupakan fitur dari Apache Lucene yang dapat menghasilkan query untuk mencari dokumen yang memiliki atribut yang serupa dengan dokumen tertentu.

Dalam MoreLikeThis, terdapat beberapa konfigurasi yang dapat disesuaikan untuk meningkatkan kualitas hasil pencarian. Berikut daftar konfigurasi yang dapat diatur pada MoreLikeThis :

- setMinTermFreq(tf minimum)
- setMinDocFreq(df minimum)
- setMaxDocFreq(df maksimum)
- setMinWordLen(panjang kata minimum)
- setMaxWordLen(panjang kata maksimum)
- setMaxQueryTerms(panjang query maksimum)
- setMaxNumTokensParsed(jumlah token maksimum)

Halaman ini sengaja dikosongkan

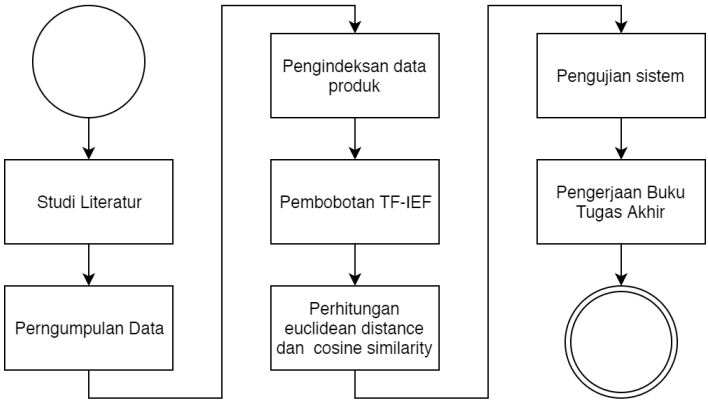
BAB 3

METODOLOGI

Pada bab metodologi akan menjelaskan bagaimana langkah pengerjaan tugas akhir dengan disertakan deskripsi dari setiap penjelasan untuk masing-masing tahapan beserta jadwal kegiatan pengerjaan tugas akhir.

3.1 Tahapan pengerjaan tugas akhir

Tahapan pelaksanaan tugas akhir yang menjelaskan mengenai proses pengerjaan gambar 3.1



Gambar 3.1: Metodologi Tugas Akhir

3.1.1 Studi literatur

Dalam studi literatur, konsep dan teori yang berkaitan dengan pengerjaan tugas akhir seperti linked data, perhitungan entity similarity akan dipelajari terlebih dahulu. Penelitian sebelumnya yang terkait dengan tugas akhir juga akan dipeajari untuk mencari gap dan peluang yang dapat yang dapat dikembangkan dari penelitian sebelumnya.

3.1.2 Pengumpulan data

Data yang tercatat di aplikasi Halal Nutrition Food saat ini masih berjumlah 102 produk. Data akan ditambah dengan melakukan web scrapping pada situs pencarian produk halal LPPOM MUI sehingga informasi produsen, nama produk, tanggal berlaku sertifikat bisa ditambahkan. Untuk informasi komposisi produk akan ditambahkan secara manual mengingat belum adanya sumber yang dapat digunakan sebagai acuan dari produk halal yang sudah tercatat di aplikasi Halal Nutrition Food. Untuk membantu memudahkan proses entri komposisi produk digunakan sistem crowdsourcing dengan gamifikasi

3.1.3 Pengindeksan data produk

Data yang telah dikumpulkan tersebut kemudian akan diindex ke dalam Apache Lucene. Pengindeksan ini dilakukan untuk menyimpan term dari dokumen ke dalam index Lucene, sehingga dari index tersebut akan dapat dilakukan perhitungan bobot menggunakan TFIEF yang telah dijelaskan pada bab 2.2.3.

Sebelum data produk diindex, data harus diolah terlebih dahulu.

Pre-processing data dilakukan untuk mempercepat pembobotan dan perhitungan kemiripan antar produk. Pada tugas akhir ini pre-processing yang dilakukan adalah tokenisasi, yaitu pemotongan kalimat menjadi kata penyusunnya, yang disebut token, berdasarkan spasi dan tanda baca. Token ini yang nantinya akan diindex ke menggunakan Apache Lucene.

3.1.4 Pembobotan TF-IEF

Data produk beserta komposisi yang telah tercatat kemudian dihitung pembobotannya menggunakan tf-ief. Pembobotan tf-ief akan membandingkan komposisi dari setiap produk dengan komposisi produk lain dari seluruh produk yang telah tercatat. Pembobotan dibantu menggunakan Apache Lucene dengan membaca index yang telah dibuat sebelumnya.

Sebagai contoh pembobotan untuk produk ”Indomie Mie Goreng”. Term yang terdapat dalam produk Indomie Mie Goreng adalah sebagai berikut :

- Vegetable
- Oil
- Salt
- Wheat
- Flour
- Sweet
- Soy
- Sauce
- Chili

Pertama dihitung nilai tf yang dimiliki term (komposisi) dari dokumen (produk). Nilai tf merupakan jumlah berapa kali term muncul di dalam produk dibagi dengan jumlah term pada produk (panjang

dokumen). Total produk saat perhitungan ini dilakukan adalah 102. Berikut tabel 3.1 dan tabel 3.2 berisikan perhitungan tf dan bobot tf-ief untuk *term* yang terdapat pada produk Indomie Mie Goreng dan Po Mie Rasa Soto Ayam:

Term	Tf	Tf-Ief
Vegetable	1	0.189
Oil	1	0.189
Salt	1	0.189
Wheat	1	0.189
Flour	1	0.189
Sweet	1	0.189
Soy	1	0.189
Sauce	2	0.340
Chili	1	0.189

Tabel 3.1: Bobot tf-ief produk Indomie Mie Goreng

Term	Tf	Tf-ief
Sugar	1	0.155
Salt	1	0.155
Vegetable	1	0.155
Oil	1	0.155
Wheat	1	0.155
Flour	1	0.155
Butylated	1	0.155
hydroxyanisole	1	0.155
Tartrazine	1	0.155
Monosodium	1	0.155
glutamate	1	0.155

Tabel 3.2: Bobot tf-ief produk Pop Mie Rasa Soto Ayam

3.1.5 Perhitungan euclidean distance

Perhitungan euclidean distance memperhitungkan jarak antara 2 titik di dalam ruang euclidean. Dokumen dibobotkan terlebih dahulu menggunakan tf-ief untuk merepresentasikannya sebagai titik di dalam ruang euclidean.

Tabel berikut menunjukkan langkah-langkah dalam perhitungan euclidean distance antara produk Indomie Mie Goreng (d_1) dan Pop Mie Rasa Soto Ayam (d_2) :

Term	w_{d1}	w_{d2}	$(w_{d1} - w_{d2})^2$
Vegetable	0.189	0.155	0.001156
Oil	0.189	0.155	0.001156
Salt	0.189	0.155	0.001156
Wheat	0.189	0.155	0.001156
Flour	0.189	0.155	0.001156
Sweet	0.189	0	0.035721
Soy	0.189	0	0.035721
Sauce	0.34	0	0.1156
Chili	0.189	0	0.035721
Sugar	0	0.155	0.024025
Butylated	0	0.155	0.024025
hydroxyanisole	0	0.155	0.024025
Tartrazine	0	0.155	0.024025
Monosodium	0	0.155	0.024025
glutamate	0	0.155	0.024025

TOTAL			0.372693
SQRT			0.610485872

Tabel 3.3: Langkah-langkah perhitungan euclidean distance produk Indomie Mie Goreng dan Pop Mie Rasa Soto Ayam

Berdasarkan bab 2.2.4, similarity dalam euclidean distance dapat dituliskan sebagai

$$sim(p,q) = \frac{1}{dist(p,q)+1} \tag{3.1}$$

Maka didapatkan :

$$sim(p,q) = \frac{1}{0.610485872+1} = 0.620930626 \tag{3.2}$$

3.1.6 Perhitungan cosine similarity

Perhitungan cosine similarity menggunakan pembobotan yang telah dihitung menggunakan tf-idf sebelumnya. Perhitungan cosine similarity dibantu menggunakan Apache Lucene yang hasilnya akan ditampilkan ke pengguna.

Tabel di bawah menujukkan langkah-langkah perhitungan dalam cosine similarity antara produk Indomie Mie Goreng (d_1) dan Pop Mie Rasa Soto Ayam (d_2) :

Term	w_{d1}	w_{d2}	$w_{d1} \times w_{d2}$	w_{d1}^2	w_{d2}^2
Vegetable	0.189	0.155	0.029	0.036	0.024

Oil	0.189	0.155	0.029	0.036	0.024
Salt	0.189	0.155	0.029	0.036	0.024
Wheat	0.189	0.155	0.029	0.036	0.024
Flour	0.189	0.155	0.029	0.036	0.024
Sweet	0.189	0	0	0.036	0
Soy	0.189	0	0	0.036	0
Sauce	0.34	0	0	0.116	0
Chili	0.189	0	0	0.036	0
Sugar	0	0.155	0	0	0.024
Butylated	0	0.155	0	0	0.024
hydroxyanisole	0	0.155	0	0	0.024
Tartrazine	0	0.155	0	0	0.024
Monosodium	0	0.155	0	0	0.024
glutamate	0	0.155	0	0	0.024
TOTAL			0.146	0.401	0.264
SQRT				0.634	0.514

Tabel 3.4: Langkah-langkah perhitungan cosine similarity produk Indomie Mie Goreng dan Pop Mie Rasa Soto Ayam

Selanjutnya hasil perhitungan dimasukkan ke dalam formula cosine similarity :

$$Sim(d_1, d_2) = \frac{\sum_w weight(w, d) * weight(w, d')}{\sqrt{\sum_w weight(w, d)^2}\sqrt{\sum_w weight(w, d')^2}} \tag{3.3}$$

$$Sim(d_1, d_2) = \frac{0.146}{0.634 \times 0.514} = 0.450 \tag{3.4}$$

3.1.7 Pengujian sistem

Pada tahap ini dilakukan pengujian setelah sistem selesai dibuat. Pengujian dilakukan untuk mengetahui apakah terjadi error pada saat menjalankan program dan apakah hasil dari sistem sesuai dengan yang diharapkan. engujian fungsionalitas mengikuti skenario pengujian yang telah dibuat sebelumnya. Selain melakukan pengujian dari algoritma cosine dan euclidean similarity, juga dilakukan pengujian dari fitur MoreLikeThis yang dimiliki Apache Lucene. Berikut adalah skenario pengujian fungsionalitas untuk aplikasi Halal Nutrition Food :

No.	Faktor	Detil	Ukuran
1.	Ketepatan	Perhitungan kemiripan menggunakan Euclidean distance	Presisi
2.	Ketepatan	Perhitungan kemiripan menggunakan Cosine similarity	Presisi
3.	Ketepatan	Perhitungan kemiripan menggunakan MoreLikeThis	Presisi

Tabel 3.5: Skenario Pengujian Fungsional

3.1.8 Pengerjaan buku tugas akhir

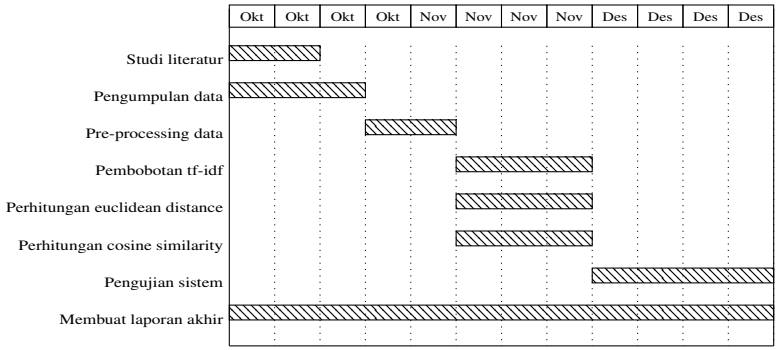
Pada tahap ini, seluruh pengerjaan tugas akhir dari awal hingga akhir didokumentasikan ke dalam sebuah buku tugas akhir. Buku tugas akhir dibuat berdasarkan panduan penyusunan buku tugas akhir yang telah ditentukan.

3.2 Penjelasan sistem

Aplikasi Halal Nutrition Food menggunakan Apache sebagai web server dan juga MySQL sebagai basis datanya. Aplikasi Halal Nutrition Food dibangun menggunakan framework Laravel. Selain itu juga digunakan Apache Lucene untuk melakukan pembobotan data , perhitungan euclidean distance dan cosine similarity. Pengguna dapat melakukan pencarian produk halal dan juga mengentri produk baru. Saat produk baru dientri, data produk akan disimpan ke dalam basis data. Setelah data diverifikasi dan di-*generate* ke dalam file turtle, Apache Lucene kemudian mengindex data produk dan melakukan pembobotan tf-ief pada komposisi dan label dari produk yang telah dimasukkan. Setelah seluruh komposisi dari seluruh produk dibobotkan, Apache Lucene kemudian menghitung kemiripan antara produk satu dengan yang lain dengan euclidean distance dan cosine similarity. Hasil perhitungan kemiripan akan disimpan di MySQL.

Ketika pengguna memilih produk yang akan dilihat, aplikasi Halal Nutrition Food akan melakukan query ke basis data untuk mengambil informasi nama produk, produsen, nomor sertifikat produk dan komposisinya. Aplikasi kemudian akan mengambil lima informasi produk dengan kemiripan terbesar dari database MySQL dan ditampilkan ke dalam website.

3.3 Jadwal kegiatan



BAB 4

PERANCANGAN

Pada bab ini membahas terkait alur perancangan terkait beberapa hal yang diperlukan dalam proses pembuatan aplikasi sesuai dengan alur yang dijelaskan pada bab 3. Dalam bab perancangan ini akan menjelaskan tentang proses penggalian kebutuhan dan desain sistem.

4.1 Pengumpulan Data

Pada tahap ini dimasukkan data produk ke dalam aplikasi Halal Nutrition Food. Data sertifikasi diambil dari MUI sedangkan data komposisi diambil dari kemasan produk. Pengumpulan data juga dilakukan secara *crowdsourcing* menggunakan fitur input produk yang terdapat pada website

4.1.1 Generating turtle

Data yang tersimpan di dalam database dikonversi menjadi file turtle (.ttl) menggunakan kode PHP. Berikut adalah contoh salah satu konten yang terdapat pada file turtle :

```
halalf : Happy_Tos_Rasa_Jagung_Bakar a halalv : FoodProduct;
        halalv : foodCode "8993027163754";
        rdfs : label  "Happy Tos Rasa Jagung Bakar";
        halalv : manufacture "PT. Sinar Kencana Agung";
        halalv : netWeight 55;
        halalv : calories 280;
        halalv : fat 14;
```

```
halalv : saturatedFat  6;
halalv : sodium 99.9;
halalv : fiber  4;
halalv : sugar  1;
halalv : protein  4;
halalv : vitaminA 0;
halalv : vitaminC 0;
halalv : calcium  0;
halalv : iron  0;
halalv : foodproductId  2.
halalv : manufacture  halalm :2.
halalv :  certificate   halalc :2.
halalv :  containsIngredient   halali : Whole_Corn,  halali :
    Palm_Oil,  halali : Monosodium_glutamate,  halali :
    Flavour_Enhacer,  halali : Sunset_Yellow_FCF.
```

4.2 Desain Index Lucene

Data produk yang sudah diubah menjadi file turtle akan diindex untuk mengetahui dan menyimpan nilai *term frequency, document frequency* dan bobot dari tiap *term* yang terdapat di dalam file turtle.

Konten dari file turtle akan dibaca dan diindex. Atribut dari index yang akan digunakan dalam pembobotan TF-IEF untuk mencari produk yang terkait adalah :

1. label
2. containsIngredient

Hal ini didasarkan pada hasil survei pada penelitian sebelumnya[3], yang menunjukkan atribut dengan bobot tertinggi dalam pencarian produk.

4.2.1 Pre-processing Data

Untuk mengindex file turtle, digunakan library Apache Jena yang dapat membaca nilai subjek, predikat, objek dari dokumen turtle.

Berikut adalah contoh hasil pembacaan file turtle menggunakan Apache Jena :

```
predikat : http://halal.addi.is.its.ac.id/halalv.ttl#containsIngredient
objek    : http://halal.addi.is.its.ac.id/resources/ingredients/
           Monosodium_glutamate

predikat : http://www.w3.org/2000/01/rdf-schema#label
objek    : Happy Tos Rasa Jagung Bakar

predikat : http://halal.addi.is.its.ac.id/halalv.ttl#netWeight
objek    : 55(ttp://www.w3.org/2001/XMLSchema#integer
```

Hasil pembacaan predikat dan objek dari file turtle masih berupa URI. Nilai untuk predikat dan objek kemudian diambil memotong elemen yang terdapat pada URI. Berikut adalah hasil pembacaan setelah pembacaan predikat dan objek diambil dari potongan URI :

```
predikat :  containsIngredient
objek    : Monosodium glutamate

predikat : label
objek    : Happy Tos Rasa Jagung Bakar

predikat : netWeight
objek    : 55
```

Objek yang telah dibersihkan kemudian ditokenisasi dan disimpan ke dalam index. Berikut adalah skema index Lucene yang siap digunakan untuk mencari produk terkait seperti pada gambar 4.1

Doc : Happy Tos Rasa Jagung Bakar

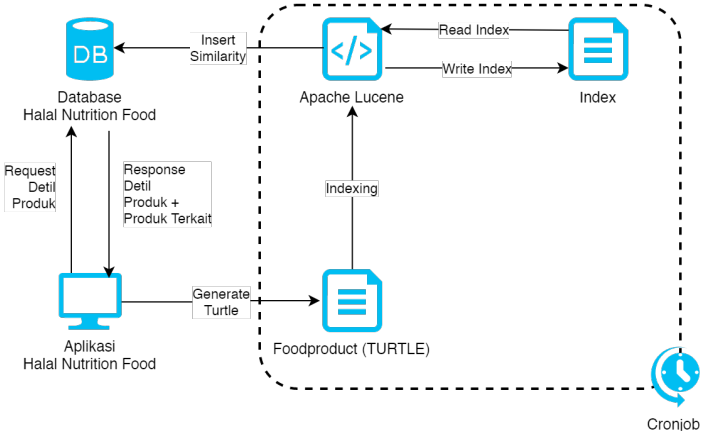
label
bakar;happy;jagung;rasa;tos;

containsIngredient
corn;enhacer;fcf;flavour;glutamate;monosodium;oil;
palm;sunset;whole;yellow;

Gambar 4.1: Desain index Lucene

4.3 Desain Sistem

Sistem pencarian produk terkait pada aplikasi Halal Nutrition Food ini terbagi ke dalam 2 proses, yaitu proses pengindeksan dan proses query produk terkait seperti pada gambar 4.2



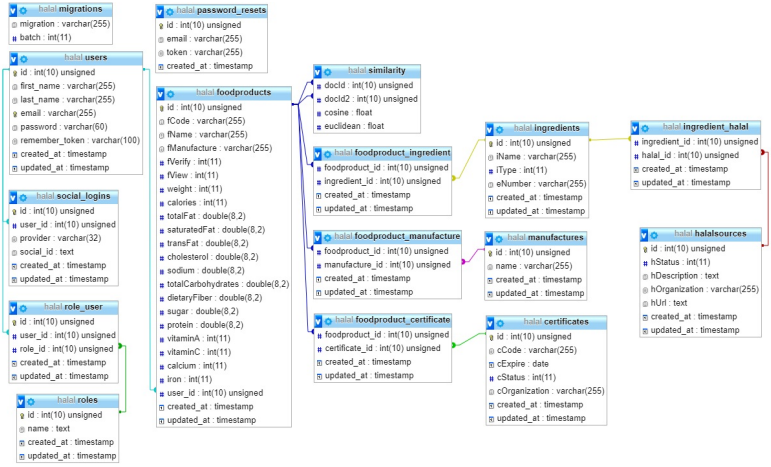
Gambar 4.2: Desain sistem dalam aplikasi

Penjelasan desain sistem:

1. Proses Indexing, proses ini dilakukan oleh Apache Lucene yaitu dengan membaca dan mengambil *field* dari file turtle beserta nilai *term frequency*-nya. Nilai-nilai tersebut kemudian disimpan ke dalam file index. File index kemudian dibaca lagi oleh Apache Lucene untuk membandingkan semua produk yang telah diindeks dengan menghitung skor nilai cosine similarity dan educlidean distance berdasarkan bobot TF-IEF yang diperoleh dari nilai *term ferquency* dari *field* yang telah diindex. Nilai cosine similarity dan educlidean distance yang telah dihitung kemudian disimpan ke dalam tabel similarity pada basis data MySQL. Proses pengindeksan dilakukan dengan bantuan cronjob yang akan dieksekusi 1 bulan sekali.
2. Proses query produk terkait, proses ini berjalan ketika user hendak melihat detil dari produk yang terdaftar dalam aplikasi Halal Nutrition Food. Aplikasi Halal Nutrition Food akan mengambil produk terkait lainnya berdasarkan id produk yang tersimpan pada tabel similarity dan menampilkannya kepada user urut berdasarkan kemiripannya.

4.4 Desain Basis Data

Skema basis data dikembangkan dari skema basis data aplikasi Halal Nutrition Food sebelumnya [3], dengan penambahan tabel similarity. Skema basis data yang digunakan dalam aplikasi Halal Nutrition Food ini digambarkan pada Gambar 4.3



Gambar 4.3: Skema basis data yang digunakan dalam aplikasi

Penjelasan tabel dalam skema basis data:

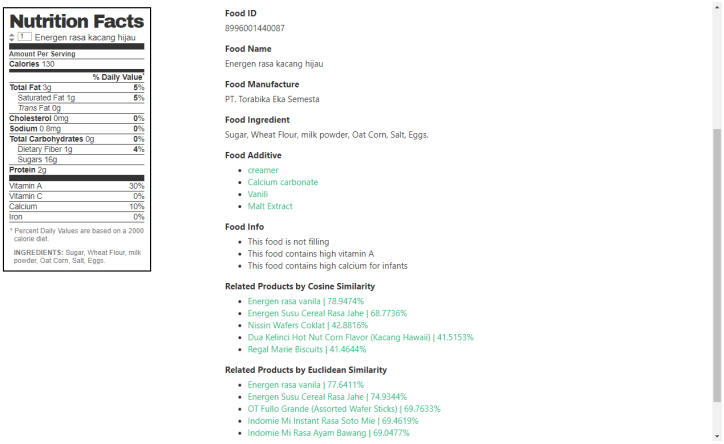
- foodproducts berisi data-data utama dari produk makanan dan minuman seperti kode produk, nama produk, produsen, kandungan gizi, dan nama pengguna yang memasukkan produk tersebut. Tabel ini menjadi referensi untuk beberaoa tabel berikutnya.
- foodproducts_ingredient adalah tabel relasi antara tabel foodproducts dan tabel ingredient. Di dalamnya berisi kolom id produk dan id ingredient yang ada di dalam setiap produk.
- foodproducts_manufacture adalah tabel relasi antara foodproducts dan tabel manufacture. Di dalamnya berisi kolom id produk dan id manufacture.
- foodproducts_certificate adalah tabel relasi antara tabel foodproducts dan tabel certificate. Di dalamnya berisi kolom id produk dan id certificate yang dimiliki oleh setiap produk.
- similarity adalah tabel yang menyimpan nilai kemiripan dua

produk. Di dalamnya berisi kolom docId dan docId2 yang berasal dari id produk, nilai *cosine* dan *euclidean similarity* antara dua produk.

- ingredients berisi data mengenai bahan-bahan atau zat aditif yang terkandung dalam setiap produk. Di dalamnya berisi kolom nama zat, tipe zat, eNumber zat, dsb.
- ingredients_halal adalah tabel relasi antara tabel ingredients dengan tabel halalsources. Di dalamnya berisi kolom id ingredients dan id dari sumber halal. Fungsinya untuk melihat apakah bahan tersebut termasuk bahan yang halal atau tidak.
- halalsources berisi data mengenai penjelasan sumber halal.
- manufactures berisi data mengenai produsen makanan dan minuman tersebut.
- certificates berisi data mengenai sertifikat halal yang dimiliki oleh sebuah produk dari berbagai sumber.
- users berisi data pengguna yang telah terdaftar dalam aplikasi yang merupakan tabel referensi untuk menentukan role dari setiap pengguna.
- roles berisi data mengenai role pengguna yang ada di dalam aplikasi.
- role_user adalah tabel relasi antara tabel users dan tabel roles. Di dalamnya berisi kolom id user dan id role. Fungsinya untuk menjelaskan seorang pengguna mempunyai role sebagai apa.
- social_login adalah tabel yang berisi data mengenai pengguna yang login ke aplikasi menggunakan sosial media yang dimilikinya.
- migrations adalah tabel bawaan dari Laravel yang berfungsi untuk menyimpan *migration* yang dimiliki oleh aplikasi.

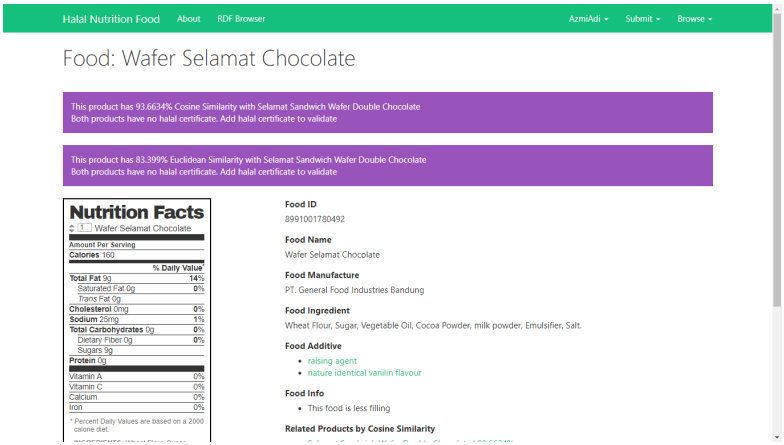
4.5 Desain Antarmuka Pengguna

Tugas akhir ini berfokus untuk menampilkan produk-produk yang terkait dengan produk yang sedang dilihat oleh pengguna. Oleh karena itu terdapat penambahan item produk terkait pada halaman detil produk.



Gambar 4.4: Halaman produk dengan produk terkait

Halaman produk yang digambarkan pada Gambar 4.4 ini tampil ketika pengguna memilih produk dari daftar hasil pencarian dari halaman sebelumnya. Di dalam halaman ini berisi informasi lengkap mengenai sebuah produk makanan dan minuman, seperti nama produk, kode produk, produsen, bahan zat aditif, sertifikat halal, kandungan gizi dan produk yang terkait dengan produk tersebut.



Gambar 4.5: Notifikasi pada halaman detil produk

Pada halaman detil produk juga akan ditampilkan notifikasi kepada user apabila terdapat produk yang memiliki kemiripan di atas 75% dengan produk lain seperti pada gambar 4.5 . Pemberitahuan juga akan ditampilkan apabila terdapat produk yang belum tersertifikasi dan mengandung bahan aditif dalam kategori haram.

Halaman ini sengaja dikosongkan

BAB 5

IMPLEMENTASI

Pada bab ini akan dijelaskan terkait proses implementasi pada perangkat lunak yang dirancang.

5.1 Lingkungan Implementasi

Pada bagian ini dibahas terkait lingkungan pengujian yang digunakan dalam implemetasi tugas akhir terkait perangkat yang digunakan baik perangkat keras maupun perangkat lunak. Tabel 5.1 berisikan spesifikasi perangkat keras untuk implementasi pada tugas akhir ini.

Tabel 5.1: Spesifikasi Perangkat Keras

Perangkat	Spesifikasi
Jenis	ASUS A450 CA
Processor	Intel Celeron 1007U
RAM	6GB
Hard Disk Drive	500GB

Kemudian untuk perangkat lunak yang digunakan dalam implementasi model ditunjukkan dalam tabel 5.2.

Tabel 5.2: Spesifikasi Perangkat Lunak

Nama Perangkat Lunak	Kegunaan dalam Implementasi
Xampp 7.0.15 dengan PHP 5.6.14	Webserver
Apache Lucene	Search Engine
Netbeans 8.2	Java IDE

Notepad++	Text Editor
Google Chrome 62	Web Browser

5.2 Implementasi

Pada proses implementasi akan dilakukan realisasi dari perancangan yang sudah dibuat sebelumnya. Berikut adalah proses implementasi yang dilakukan:

1. Konfigurasi server
2. Pembacaan file turtle
3. Pengindeksan data produk
4. Pembacaan index
5. Pembobotan data produk
6. Perhitungan cosine similarity dan euclidean
7. Pembuatan query untuk menampilkan produk terkait

5.2.1 Konfigurasi server

Konfigurasi server yang dimaksud meliputi konfigurasi untuk Apache Web Server dan MySQL.

- Apache Web Server berjalan pada port 80 sebagai server dari aplikasi Halal Nutrition Food.
- MySQL berjalan pada port 3306 sebagai database yang menampung data produk makanan, komposisi, produk terkait, dan konfigurasi aplikasi Halal Nutrition Food.
- Java Runtime Environment (JRE) yang terinstal adalah versi 1.8.0_151.

5.2.2 Pembacaan file turtle

File turtle dibaca menggunakan library dari Apache Jena. Konfigurasi untuk pembacaan file turtle yang digunakan adalah sebagai berikut :

```
/*membuat model yang membaca inputstream*/
InputStream in = Files.newInputStream(path);
Model model = ModelFactory.createDefaultModel();
model.read(in, null, "TURTLE");

/* arraylist untuk menyimpan subjek, predikat, objek sementara*/
ArrayList<TurtleModel> turtle = new ArrayList<>();

/*query terhadap turtle */
String query = ""
        + "SELECT ?p ?o WHERE { "
        + "?s ?p ?o."
        + "}";

QueryExecution exec = QueryExecutionFactory.create(query, model);
ResultSet rs = exec.execSelect();

while (rs.hasNext()) {

    QuerySolution qs = rs.next();

    String p = qs.get("p").toString();
    String o = qs.get("o").toString();

}
```

Konfigurasi tersebut akan membaca subjek, predikat, objek file turtle dari inputStream yang berupa path yang merujuk pada file turtle.

Selanjutnya, masih di dalam blok *while*, untuk melakukan pre-processing

seperti yang telah dijelaskan pada bab 4.2.1, berikut potongan kode untuk mengambil potongan elemen yang ada pada URI dari file turtle :

```
/*mengambil predikat setelah di akhir prefix */
String p = qs.get("p").toString();
String pred[] = p.split("#");
String predikat = pred[pred.length - 1];

/*mengambil dan membersihkan objek*/
String o = qs.get("o").toString();
String objek = o;
if (o.contains("^^")) {
    String obj[] = o.split("\\^\\^");
    objek = obj[0];
} else if (o.contains("/")) {
    String obj[] = o.split("/");
    objek = obj[obj.length - 1];
    objek = objek.replace("halalv.ttl#", "");
} else {
    objek = o;
}

if (objek.contains("_")) {
    objek = objek.replace("_", "␣");
}

/*menambahkan nilai subjek, predikat, objek ke arraylist */
turtle.add(predikat, objek);
```

Konfigurasi tersebut akan menghilangkan bagian URI yang tidak dibutuhkan dalam proses indexing dan hanya nilai dari masing-masing subjek, predikat, dan objek.

Nilai subjek, predikat, objek yang sudah bersih kemudian ditambahkan ke dalam ArrayList<TurtleModel>untuk disimpan sementara untuk kemudian ditambahkan ke dalam index yang akan dije-

laskan pada sub bab 5.2.3.

Berikut kelas TurtleModel yang digunakan untuk menyimpan nilai subjek, predikat, dan objek :

```
public class TurtleModel {

    private final String predicate ;
    private final String object ;

    public TurtleModel( String predicate , String object ) {
        this . predicate = predicate ;
        this . object = object ;
    }

    public String getPredicate () {
        return predicate ;
    }

    public String getObject () {
        return object ;
    }
}
```

5.2.3 Pengindeksan data produk

Data produk diindex menggunaan Apache Lucene agar dapat dibaca kembali untuk menghitung kemiripan produk menggunakan cosine similarity dan euclidean berdasarkan bobot dari tiap produk. Perhitungan bobot akan dijelaskan pada sub bab selanjutnya.

Berikut adalah konfigurasi yang digunakan untuk melakukan indexing terhadap field dari file turtle dengan bantuan Apache Lucene :

```
FieldType  TYPE_STORED = new FieldType();

TYPE_STORED.setTokenized(true);
TYPE_STORED.setStored(true);
TYPE_STORED.setStoreTermVectors(true);
TYPE_STORED.setIndexOptions(IndexOptions.DOCS_AND_FREQS);
TYPE_STORED.freeze();
```

Konfigurasi tersebut merupakan konfigurasi untuk *field* dari tiap dokumen berdasarkan desain index pada bab 4.2. Konfigurasi tersebut akan meminta agar proses indexing melakukan tokenisasi pada field, menyimpan nilai dari field, dan menyimpan *term vector* dari field

Term vector merupakan nilai dari field yang telah ditokenisasi beserta jumlah kemunculan term (*term frequency*) di dalam field tersebut.

Berikut adalah contoh term vector pada field label, yaitu ”Happy Tos Rasa Jagung Bakar” dapat dilihat pada tabel 5.3

Tabel 5.3: Contoh term vector pada label ”Happy Tos Rasa Jagung Bakar”

Term	Freq.
bakar	1
happy	1
jagung	1
rasa	1
toast	1

Untuk membuat index, ditentukan terlebih dahulu direktori penyimpanan index dan dibuat instance dari IndexWriter.

IndexWriter akan menuliskan index ke dalam direktori sesuai dengan konfigurasi yang diatur dalam IndexWriterConfig. Berikut adalah inisiasi dari IndexWriter dan IndexWriterConfig :

```
/*menentukan direktori index*/
Directory directory = FSDirectory.open(Paths.get(INDEX_PATH));
Analyzer analyzer = new StandardAnalyzer();

/*membuat instance IndexWriterConfig*/
IndexWriterConfig iwc = new IndexWriterConfig(analyzer);
iwc.setSimilarity (new TFIEFSimilarity());

/*membuat instance IndexWriter*/
IndexWriter writer = new IndexWriter( directory , iwc);
```

Pada IndexWriterConfig, digunakan StandardAnalyzer untuk proses analisa dan tokenisasi teks sesuai dengan Work Break Rules (http://unicode.org/reports/tr29/) yaitu melakukan tokenisasi berdasarkan spasi dan tanda baca.

Pada IndexWriterConfig juga diatur similarity yang digunakan yaitu TFIEFSimilarity, seperti yang telah dijelaskan pada ”bab sebelumnya”. Implementasi TFIEFSimilarity dilakukan dengan meng-*extend* kelas Similarity dari Apache Lucene yang akan dibahas pada sub bab selanjutnya.

Konfigurasi indexing terhadap field digunakan saat menambahkan field ke dalam dokumen index.

Setelah semua field dimasukkan ke dalam dokumen index, dilanjutkan dengan pembuatan index. Berikut adalah konfigurasi untuk memasukkan field ke dalam dokumen index :

```
Document doc = new Document();

for (int i = 0; i < turtle.size(); i++) {
```

```
    Field field = new Field(
        turtleModels.get(i).getPredicate(),
        turtleModels.get(i).getObject(),
        TYPE_STORED
    );

    doc.add(field);
}

writer.addDocument(doc);
```

Variabel predikat dan objek merupakan hasil dari pembacaan file turtle yang telah dijelaskan pada bab 5.2.2.

5.2.4 Pembacaan index

Setelah index dari seluruh dokumen berhasil dibuat dan disimpan, Apache Lucene akan melakukan iterasi untuk membaca dan membandingkan kemiripan suatu produk dengan seluruh produk lainnya yang telah diindex. Berikut adalah baris kode yang digunakan untuk membaca dan membandingkan seluruh produk dalam index :

```
//menyimpan sementara term dari dokumen ke-i dan j
private Set<String> terms = new HashSet<>();

Directory directory = FSDirectory.open(Paths.get(INDEX_PATH));
IndexReader reader = DirectoryReader.open(directory);

for (int i = 0; i < reader.numDocs(); i++) {

    //menyimpan nilai cosine dan euclidean similarity
    // dari dokumen ke-i dan j

    ArrayList<SimilarityModel> similarity = new ArrayList<>();
```

```
for (int j = 0; j < reader.numDocs(); j++) {

    // map untuk menyimpan term, tf, df, dan N
    Map<String, Integer> tf;
    Map<String, Integer> tf2;

    //Pembobotan data produk
    //simpan term, tf, df, dan N dari dokumen ke-i dan j
    tf = getTf(reader, i);
    tf2 = getTf(reader, j);

    //map untuk menyimpan term dan bobot gabungan dua produk
    Map<String,Double> v1 = mapToMap(tf);
    Map<String,Double> v2 = mapToMap(tf2);

    /*Perhitungan cosine similarity dan euclidean*/
    double sim_cosine;
    double sim_euclid;
    sim_cosine = getCosineSimilarity (v1, v2);
    sim_euclid = getEuclideanSimilarity (v1, v2);

    //menambahkan similarity ke arraylist
    int id1 = reader.document(i).get("foodproductId");
    int id2 = reader.document(j).get("foodproductId");

    similarity .add(
            new SimilarityModel(
                    id1,
                    id2,
                    sim_cosine,
                    sim_euclid
            )
    );

    /* Insert similarity ke MySQL*/
    int n = 5; //ambil 5 similarity teratas
```

```
            insertTopSimilarity ( similarity , n, connection,
                sortbyCosine);
            insertTopSimilarity ( similarity , n, connection,
                sortbyEuclid);
        }
}
```

Direktori penyimpanan index yang telah dibuat sebelumnya dibuka terlebih dahulu kemudian dibuatkan instance dari IndexReader. IndexReader akan membaca direktori index yang telah dibuka sebelumnya.

Untuk membandingkan seluruh produk satu dengan yang lain, dilakukan dua iterasi bertingkat dari tiap dokumen yang terbaca di dalam direktori index. Untuk setiap dokumen dalam iterasi akan dikirim sebagai parameter ke fungsi getTf() agar nilai *tf* dari term dokumen dapat dibaca dari index. Berikut adalah baris kode dari fungsi getTf() :

```
private Map<String, Integer> getTf(IndexReader reader, int docId)
    throws IOException {

    int docCount = reader.numDocs();
    Map<String, Integer> map = new HashMap<>();

    /* membuat iterator untuk term vector
    dari field " containsIngredient " */

    Terms vector = reader.getTermVector(docId, "
        containsIngredient ");
    TermsEnum termsEnum = vector.iterator();
    BytesRef text = null;

    while ((text = termsEnum.next()) != null) {
        if (termsEnum.seekExact(text)) {
            String term = text.utf8ToString();
```

```
            PostingsEnum docs = termsEnum.postings(null,
                PostingsEnum.FREQS);
            int tf = 0;

            while (docs.nextDoc() != DocIdSetIterator.
                NO_MORE_DOCS) {

                tf = docs.freq();

            }
            //menambahkan termStat ke Map
            map.put(term, tf);

            //menambahkan term ke Set
            terms.add(term);
        }
    }

    /*membuat iterator untuk term vector dari field "label"*/

    Terms vector2 = reader.getTermVector(docId, "label");
    TermsEnum termsEnum2 = vector2.iterator();
    BytesRef text2 = null;

    while ((text2 = termsEnum2.next()) != null) {
        if (termsEnum2.seekExact(text2)) {
            String term2 = text2.utf8ToString();

            PostingsEnum docs2 = termsEnum2.postings(null,
                PostingsEnum.FREQS);
            int tf = 0;

            while (docs2.nextDoc() != DocIdSetIterator.
                NO_MORE_DOCS) {

                tf = docs2.freq();
```

```
            }

            //cek apakah terms sudah ada di Ingredients
            if (map.containsKey(term2)) {

                //ambil term stat dari ingredient
                int savedTf = map.get(term2).getTf();

                //ambil term stat dari label
                int currentTf = termStats2.getTf() ;;

                //hitung tf dan df gabungan
                int mergedTf = savedTf + currentTf;

                //menambahkan termStat baru ke Map
                map.remove(term2);
                map.put(term2, mergedTf);

            } else {
                //menambahkan termStat ke Map
                map.put(term2, tf);

                //menambahkan term ke Set
                terms.add(term2);
            }
        }
    }
    return map;
}
```

Fungsi getTf() menerima parameter berupa IndexReader dan id dokumen yang didapat dari iterasi bertingkat sebelumnya. Fungsi getTf() akan mengembalikan objek Map<String, Integer>. Map berisi frekuensi term dari setiap term yang terdapat dalam dokumen.

5.2.5 Pembobotan data produk

Term vector dari dokumen kemudian dibaca. Term vector dari field ”containsIngredient” diambil terlebih dahulu. Nilai *N* dapat langsung diambil dari IndexReader. Kemudian untuk setiap term dalam term vector, diambil frekuensi dari term kemudian ditambahkan ke dalam Map<String, Integer>dengan term sebagai *key* dan nilai frekuensi term sebagai *value*.

Setelah menambahkan nilai frekeunsi term dari field ”containsIngredient” ke dalam map, term ditambahkan ke dalam HashSet, dan dilanjutkan dengan mengambil term vector dari field ”label”. Seperti pada field ”containsIngredient”, untuk setiap term dalam term vector, diambil nilai frekuensi term.

Sebelum nilai frekuensi term dari field ”label” ditambahkan ke dalam map, dilakukan pemeriksaan terhadap map, apakah term yang diambil dari field ”label” sudah ditambahkan dari field ”containsIngredient” sebelumnya. Apabila sudah ada, maka nilai frekuensi term dari term dijumlahkan. Apabila belum ada, maka dapat ditambahkan seperti biasa. Term dari field ”label” juga ditambahkan lagi ke dalam HashSet.

5.2.6 Perhitungan cosine similarity dan euclidean

Dalam menghitung cosine similarity dan euclidean similarity menggunakan perhitungan yang telah dijelaskan pada bab 3.1.5 dan 3.1.6, semua term yang muncul dalam field ”containsIngredient” dan ”label” dihitung bobotnya untuk setiap dokumen.

Objek Map<String, Integer>yang dikembalikan dari fungsi getTf() menyimpan term dan nilai frekuensi term dari masing-masing

dokumen. Untuk menggabungkan term dan bobot dari dokumen lain yang menjadi pembanding, maka dua objek map masing-masing dikirim ke fungsi maptoMap() sebagai parameter dan dikembalikan dalam objek Map<String, Double>yang berisi term dan bobot dari dua dokumen. Berikut adalah baris kode dari fungsi maptoMap() :

```
private Map<String, Double> mapToMap(Map<String, TermStats>
    map) {
    Map<String, Double> m = new HashMap<>();

    int docLen = 0;
    for (String term : terms) {
        if (map.containsKey(term)) {
            docLen += map.get(term);
        }
    }

    for (String term : terms) {
        if (map.containsKey(term)) {
            int tf = map.get(term);
            double weight = ((double) tf / (double) docLen) *
                Math.log10(docCount / (1 + (double) tf));
            m.put(term, weight);
        } else {
            m.put(term, 0.0);
        }
    }
    return m;
}
```

Dilakukan iterasi sepanjang terms yang sudah ditambahkan term dari masing-masing dokumen. Apabila objek map memiliki term dalam iterasi, nilai frekuensi term diambil dari objek map, dan dihitung bobotnya menggunakan TF-IEF. Apabila tidak, maka bobotnya diisi dengan 0. Bobot kemudian ditambahkan ke dalam objek Map<String, Double>dengan term dalam iterasi sebagai *key*.

Masing-masing hasil kembalian fungsi maptoMap() berupa objek Map<String, Double>kemudian dikirim ke fungsi getCosineSimilarity() sebagai parameter. Berikut adalah baris kode yang berisi fungsi untuk menghitung nilai cosine similarity dari objek map yang berisi term dan bobot dari dua dokumen :

```
private double getCosineSimilarity (Map<String, Double> v1, Map
    <String, Double> v2) {
    double dotProduct = 0;
    double sumSqW1 = 0;
    double sumSqW2 = 0;

    for (String term : terms) {
        dotProduct += v1.get(term) * v2.get(term);
        sumSqW1 += Math.pow(v1.get(term), 2);
        sumSqW2 += Math.pow(v2.get(term), 2);
    }

    return dotProduct / (Math.sqrt(sumSqW1) * Math.sqrt(
        sumSqW2));
}
```

Kode baris tersebut akan menghasilkan nilai cosine similarity sesuai dengan persamaan pada bab 3.1.6

Untuk menghitung euclidean similarity, masing-masing hasil kembalian fungsi maptoMap() berupa objek Map<String, Double>juga dikirim ke fungsi etEuclideanSimilarity() sebagai parameter. Berikut adalah baris kode yang berisi fungsi untuk menghitung nilai euclidean similarity dari objek map yang berisi term dan bobot dari dua dokumen :

```
private double getEuclideanSimilarity (Map<String, Double> v1,
    Map<String, Double> v2) {
    double sqDiff = 0;
    double euclidist = 0;
    for (String term : terms) {
```

```
            sqDiff += Math.pow((v1.get(term) - v2.get(term)), 2);
        }
        euclidist = Math.sqrt(sqDiff);
        return 1 / (euclidist + 1);
    }

        return dotProduct / (Math.sqrt(sumSqW1) * Math.sqrt(
            sumSqW2));
    }
```

Kode baris tersebut akan menghasilkan nilai cosine similarity sesuai dengan persamaan pada bab 3.1.5

Hasil perhitungan cosine dan euclidean similarity kemudian ditambahkan ke dalam ArrayList<SimilarityModel>beserta dengan foodproductId dari masing-masing dokumen yang diambil dari index. Berikut adalah baris kode dari kelas SimilarityModel :

```
public class SimilarityModel {

    private final int foodproductId, foodproductId2;
    private final double cosinesim, euclidsim;

    public SimilarityModel(int foodproductId, int foodproductId2,
            double cosinesim, double euclidsim) {
        this.foodproductId = foodproductId;
        this.foodproductId2 = foodproductId2;
        this.cosinesim = cosinesim;
        this.euclidsim = euclidsim;
    }

    public static Comparator<SimilarityModel>
        CosineComparatorDesc = new Comparator<SimilarityModel
        >() {
        @Override
        public int compare(SimilarityModel o1, SimilarityModel
            o2) {
```

```
            double cosinesim1 = o1.getCosinesim();
            double cosinesim2 = o2.getCosinesim();

            //descending
            return Double.valueOf(cosinesim2).compareTo(
                cosinesim1);
        }
    };

    public static Comparator<SimilarityModel>
        EuclideanComparatorDesc = new Comparator<
        SimilarityModel>() {
        @Override
        public int compare(SimilarityModel o1, SimilarityModel
            o2) {

            double euclidsim1 = o1.getEuclidsim();
            double euclidsim2 = o2.getEuclidsim();

            //descending
            return Double.valueOf(euclidsim2).compareTo(
                euclidsim1);
        }
    };

    public int getFoodproductId() {
        return this.foodproductId;
    }

    public int getFoodproductId2() {
        return this.foodproductId2;
    }

    public double getCosinesim() {
        return this.cosinesim;
    }
```

```
    public double getEuclidsim() {
        return this.euclidsim;
    }
}
```

ArrayList<SimilarityModel>kemudian diurutkan dan diambil berdasarkan 5 teratas untuk masing-masing cosine dan euclidean similarity. Kelima item tersebut kemudian ditambahkan ke dalam database. Struktur dari tabel yang menyimpan daftar produk terkait akan dibahas pada bab 5.2.7.

Berikut adalah baris kode untuk menambah 5 produk terkait hasil perhitungan cosine dan euclidean similarity ke dalam database :

```
int n = 5;

Comparator<SimilarityModel> sortbyCosine = SimilarityModel.
    CosineComparatorDesc;
Comparator<SimilarityModel> sortbyEuclid = SimilarityModel.
    EuclideanComparatorDesc;

 insertTopSimilarity ( similarity , n, connection, sortbyCosine);
 insertTopSimilarity ( similarity , n, connection, sortbyEuclid);
```

```
private void insertTopSimilarity (ArrayList<SimilarityModel>
    similarity, int n, Connection connection, Comparator<
    SimilarityModel> comparator) {

    //urutkan tinggi ke rendah, ambil 5 teratas
    Collections.sort( similarity , comparator);

    List<SimilarityModel> topNSimilar;
    topNSimilar = similarity .subList(0, n);

        for (int i = 0; i < topNSimilar.size(); i++) {
            insertSimilarity (
                    connection,
```

```
                    topNSimilar.get(i).getFoodproductId(),
                    topNSimilar.get(i).getFoodproductId2(),
                    topNSimilar.get(i).getCosinesim(),
                    topNSimilar.get(i).getEuclidsim()
            );
        }
    }
}

private void insertSimilarity (Connection c, int docId, int
    docId2, double cosine, double euclidean) {
    try {
        String query_insert = ""
                + "INSERT INTO similarity (docId, docId2,
                    cosine, euclidean) "
                + "VALUES (" + docId + "," + docId2 + "," +
                    cosine + "," + euclidean + ") "
                + "ON DUPLICATE KEY UPDATE " + "
                    cosine = " + cosine + "," + "euclidean = " +
                    euclidean;

        Statement st = c.createStatement();
        st.executeUpdate(query_insert);
    } catch (SQLException ex) {
        ex.printStackTrace();
    }
}
```

5.2.7 Pembuatan query untuk menampilkan produk terkait

Hasil perhitungan cosine dan euclidean similarity ditambahkan ke dalam database pada tabel 'similarity'. Berikut adalah query yang dibutuhkan untuk membuat tabel similarity :

```
--membuat tabel untuk menyimpan perhitungan similarity
```

```
CREATE TABLE ‘similarity‘ (
  ‘docId‘ int(10) UNSIGNED NOT NULL,
  ‘docId2‘ int(10) UNSIGNED NOT NULL,
  ‘cosine‘ float NOT NULL,
  ‘euclidean‘ float NOT NULL
);

--menambahkan primary key pada kolom docId dan docId2
ALTER TABLE ‘similarity‘
  ADD PRIMARY KEY (‘docId‘,‘docId2‘);

--menambahkan foreign key pada kolom docId dan docId2 terhadap id
      di tabel foodproducts
ALTER TABLE ‘similarity‘
  ADD CONSTRAINT ‘similarity_docId_foodproducts_id_foreign‘
      FOREIGN KEY (‘docId‘) REFERENCES ‘foodproducts‘ (‘id‘),
  ADD CONSTRAINT ‘similarity_docId2_foodproducts_id_foreign‘
      FOREIGN KEY (‘docId2‘) REFERENCES ‘foodproducts‘ (‘id
      ‘);
```

Setelah 5 teratas produk terkait berdasarkan cosine dan euclidean similarity ditambahkan ke dalam database, perlu dibuat query pada aplikasi web Halal Nutrition Food untuk mengambil daftar produk terkait dan menampilkannya di halaman detil produk.

Berikut adalah query yang dibutuhkan untuk mengambil 5 produk terkait berdasarkan perhitungan cosine dan euclidean similarity berdasarkan foodproductId pada halaman detil produk :

```
--membuat tabel untuk menyimpan perhitungan similarity
SELECT foodproducts.id, foodproducts.fName, similarity .cosine
FROM foodproducts
INNER JOIN similarity
ON foodproducts.id = similarity .docId2
WHERE similarity.docId = :id
ORDER BY similarity.cosine DESC
LIMIT 5
```

```
--membuat tabel untuk menyimpan perhitungan similarity
SELECT foodproducts.id, foodproducts.fName, similarity .cosine
FROM foodproducts
INNER JOIN similarity
ON foodproducts.id = similarity .docId2
WHERE similarity.docId = :id
ORDER BY similarity.euclidean DESC
LIMIT 5
```

Halaman ini sengaja dikosongkan

BAB 6

HASIL DAN PEMBAHASAN

Pada bab ini akan dijelaskan hasil dan pembahasan dari proses pengujian aplikasi.

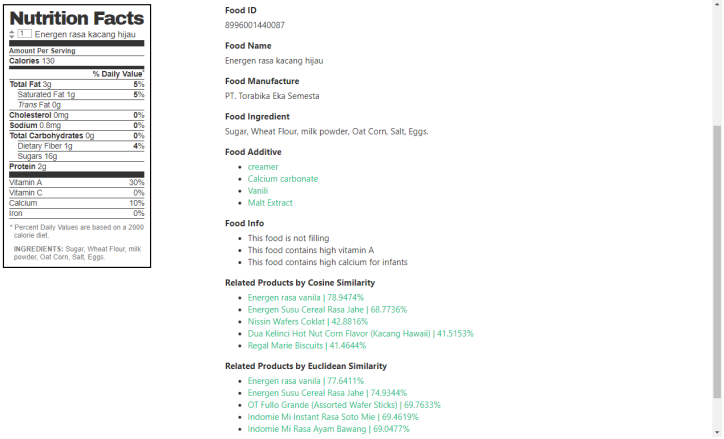
6.1 Hasil Pengujian

Pada bagian ini akan dijelaskan hasil pengujian aplikasi, baik fungsional maupun non fungsional.

6.1.1 Pengujian Algoritma Euclidean Distance dan Cosine Similarity

Pengujian algoritma cosine similarity dan euclidean distance dilakukan dengan menggunakan skenario dalam penggunaan aplikasi Halal Nutrition Food untuk mengetahui apakah aplikasi dapat menampilkan produk terkait yang sesuai dengan produk yang sedang dilihat. Berikut adalah skenario pengujian yang dilakukan :

1. Pengujian algoritma Euclidean Distance. Pada skenario ini pengujian algoritma Euclidean Distance dilakukan dengan memilih dan membuka halaman detil produk untuk melihat produk yang terkait. Dalam skenario ini detil produk yang dilihat adalah ”Energen rasa kacang hijau”. Daftar produk terkait yang ditampilkan beserta nilai cosine similarity dan euclidean distance dapat dilihat pada gambar 6.1:



Gambar 6.1: Daftar produk terkait pada produk Energen rasa kacang hijau

Berikut pada tabel 6.1 dan tabel 6.2 merupakan produk terkait dari Energen rasa kacang hijau berdasarkan perhitungan cosine similarity dan euclidean distance :

Tabel 6.1: Produk terkait ”Energen rasa kacang hijau” berdasarkan Cosine Similarity

Nama produk	Cosine Similarity
Energen rasa vanila	78.95%
Energen Susu Cereal Rasa Jahe	68.77%
Nissin Wafers Coklat	42.88%
Dua Kelinci Hot Nut Corn Flavor (Kacang Hawaii)	41.52%
Regal Marie Biscuits	41.46%

Tabel 6.2: Produk terkait ”Energen rasa kacang hijau” berdasarkan Euclidean Distance

Nama produk	Euclidean Similarity
Energen rasa vanila	77.64%

Energen Susu Cereal Rasa Jahe	74.93%
OT Fullo Grande (Assorted Wafer Sticks)	69.76%
Indomie Mi Instant Rasa Soto Mie	69.46%
Indomie Mi Rasa Ayam Bawang	69.05%

Dalam skenario ini akan dihitung nilai euclidean distance dari ”Energen rasa kacang hijau” dan ”Energen rasa vanila”. Berikut pada tabel 6.3 merupakan bobot masing-masing term dari dokumen ”Energen rasa kacang hijau” dan ”Energen rasa vanila” yang didapatkan dari hasil pembobotan term :

Tabel 6.3: TF-IEF dari term pada dokumen ”Energen rasa kacang hijau” dan ”Energen rasa vanila”

Term	Bobot Energen rasa kacang hijau	Bobot Energen rasa vanila
creamer	0.129	0.129
eggs	0.129	0
calcium	0.129	0.129
oat	0.129	0.129
salt	0.129	0.129
egg	0	0.129
vanili	0.129	0.129
corn	0.129	0.129
milk	0.129	0.129
rasa	0.129	0.129
powder	0.129	0.129
premix	0	0.129
vitamin	0	0.129
malt	0.129	0.129
extract	0.129	0.129
energen	0.129	0.129
flour	0.129	0.129
kacang	0.129	0
hijau	0.129	0
wheat	0.129	0.129
carbonate	0.129	0.129
sugar	0.129	0
vanila	0	0.129

Bobot term dari dua dokumen tersebut kemudian dimasukkan ke dalam rumus euclidean distance.

$$\sqrt{(0.129 - 0.129)^2 + (0.129 - 0)^2 + .. + (0 - 0.129)^2} \tag{6.1}$$

$$\sqrt{(0)^2 + (0.129)^2 + .. + (-0.129)^2} \tag{6.2}$$

$$\sqrt{(0 + 0.016672 + .. + 0.016672)} \tag{6.3}$$

$$\sqrt{0.133379964} = 0.365 \tag{6.4}$$

Setelah nilai euclidean distance didapatkan, maka perhitungan similarity dilakukan dengan

$$sim(p, q) = \frac{1}{dist(p, q) + 1} \tag{6.5}$$

$$sim(p, q) = \frac{1}{0.365 + 1} = 0.732 \tag{6.6}$$

2. Pengujian algoritma Cosine Similarity. Pada skenario ini pengujian algoritma Cosine Similarity dilakukan dengan memilih dan membuka halaman detil produk untuk melihat produk yang terkait. Dalam skenario ini dihitung nilai cosine similarity dari ”Energen rasa kacang hijau” dan ”Energen rasa vanila”. Bobot term diambil dari hasil perhitungan sebelumnya. Untuk menghitung cosine similarity, perhitungan dilakukan sebagai berikut :

$$\frac{(0.129 * 0.129) + (0.129 * 0) + ..(0 * 0.129)}{\sqrt{(0.129)^2 + (0.129)^2 + ...}\sqrt{(0.129)^2 + (0)^2 + ...}} \tag{6.7}$$

$$\frac{0.016672 + 0 + .. + 0}{\sqrt{0.016672 + 0.016672 + ...}\sqrt{0.016672 + 0 + ...}} \tag{6.8}$$

$$\frac{0.250087}{\sqrt{0.316777} * \sqrt{0.316777}} \tag{6.9}$$

$$\frac{0.250087}{0.562829 * 0.562829} = 0.789 \tag{6.10}$$

6.1.2 Pengujian relevansi produk terkait yang dihasilkan

Untuk menguji relevansi dari produk terkait yang ditampilkan, dilakukan pengujian presisi terhadap algoritma cosine dan euclidean distance. Selain itu, dalam skenario ini juga akan diuji algoritma yang dimiliki oleh Apache Lucene, yaitu MoreLikeThis. Konfigurasi MoreLikeThis dapat dilihat pada tabel 6.4 :

Tabel 6.4: Konfigurasi pengujian menggunakan MoreLikeThis

Fungsi	Nilai	Kerangan
setMinTermFreq	0	Mengatur jumlah minimum nilai frekuensi term yang akan diquery (default : 2)
setMinDocFreq	0	Mengatur jumlah minimum jumlah dokumen yang mengandung term yang akan diquery (default : 5)

setFieldNames	containsIngredient, label	Mengatur field yang akan dicari dalam query

Berikut adalah baris kode yang digunakan untuk mencari produk terkait menggunakan MoreLikeThis :

```
Directory directory = FSDirectory.open(Paths.get(INDEX_PATH));
IndexReader ir = DirectoryReader.open( directory );

IndexSearcher searcher = new IndexSearcher( ir );
Analyzer analyzer = new StandardAnalyzer();

String [] fields = {" containsIngredient ", "label"};

MoreLikeThis mlt = new MoreLikeThis(ir);
mlt.setAnalyzer( analyzer );
mlt.setMinTermFreq(0); //jumlah minimum term
mlt.setMinDocFreq(0); //jumlah minimum term
mlt.setFieldNames( fields ); //mengambil field "label" dan "
    containsIngredient "

for (int i = 0; i < ir.numDocs(); i++) {
    StringReader( ir.document(i).get(" containsIngredient ")) ;
    if (ir.document(i).getField ("foodproductId") != null) {

        Query query = mlt.like (i);

        TopDocs result = searcher.search(query, 6);
        ScoreDoc[] hits = result.scoreDocs;

        System.out.println ("foodproductId : " + ir.document(i).
            get("foodproductId") + ", label : " + ir.document(i
            ).get("label"));

        for (int j = 1; j < hits.length; j++) {
```

```
            Document doc = searcher.doc(hits[j].doc);
            System.out.println(j + ". foodproductId : " + doc.
                get("label") + ", score : " + hits[j].score);
        }
    }
}
```

Dalam pengujian relevansi ini, diambil beberapa produk yang memiliki produk yang berkaitan. Untuk setiap produk akan dicari produk yang terkait menggunakan algoritma cosine similarity, euclidean distance dan MoreLikeThis. Data produk yang digunakan dalam pengujian ini adalah sebagai berikut :

- Ultra Milk Plain
- Energen rasa kacang hijau
- Good Day Original
- Indomie Mi Instan Goreng
- Majorico Wafer Roll

Hasil pengujian dapat dilihat pada tabel berikut :

1. Pengujian relevansi cosine similarity

Tabel 6.5: Produk terkait "Ultra Milk Plain" berdasarkan Cosine Similarity

Nama produk	Skor	Relevansi
Ultra Milk Rasa Stroberi	72.22%	Relevan
Ultra Milk - Minuman Susu UHT Rasa Coklat	72.13%	Relevan
Indomilk	70.88%	Relevan
Nestle Bear Brand	66.65%	Relevan
Frisian Flag Pure Farm	60.58%	Relevan

Tabel 6.6: Produk terkait ”Energen rasa kacang hijau” berdasarkan Cosine Similarity

Nama produk	Skor	Relevansi
Energen rasa vanila	78.95%	Relevan
Energen Susu Cereal Rasa Jahe	68.61%	Relevan
Nissin Wafers Coklat	42.90%	Tidak
Dua Kelinci Hot Nut Corn Flavor (Kacang Hawaii)	41.77%	Tidak
Regal Marie Biscuits	41.46%	Tidak

Tabel 6.7: Produk terkait ”Good Day Original” berdasarkan Cosine Similarity

Nama produk	Skor	Relevansi
Luwak White Koffie	53.45%	Relevan
Good Day White Cappuccino	51.78%	Relevan
Good Day Tiramisu Bliss Coffee	46.12%	Relevan
Good Day Coffee Drink Avocado	41.09%	Relevan
Kapal Api White Coffee 3in1 Instant Coffee	41.01%	Relevan

Tabel 6.8: Produk terkait ”Indomie Mi Goreng Instan” berdasarkan Cosine Similarity

Nama produk	Skor	Relevansi
Indomie Mi Goreng Rasa Sate	83.92%	Relevan
Indomie Mi Rasa Kari Ayam	74.89%	Relevan
Indomie Mi Rasa Ayam Special	74.89%	Relevan
Indomie Mi Rasa Ayam Bawang	74.57%	Relevan
Indomie Mi Goreng Rendang	74.41%	Relevan

Tabel 6.9: Produk terkait ”Majorico Wafer Roll” berdasarkan Cosine Similarity

Nama produk	Skor	Relevansi
Majorico Wafer Roll Rasa Pisang Coklat	71.10%	Relevan
Astor Wafer Stick Cokelat	64.09%	Relevan

Majorico Wafer Roll Rasa Susu Vanilla	58.11%	Relevan
Pocky Chocolate Flavour	54.00%	Tidak
Wafer Selamat Chocolate	52.54%	Relevan

2. Pengujian relevansi euclidean distance

Tabel 6.10: Produk terkait ”Ultra Milk Plain” berdasarkan Euclidean Distance

Nama produk	Skor	Relevansi
Ultra Milk - Minuman Susu UHT Rasa Coklat	63.89%	Relevan
Ultra Milk Rasa Stroberi	63.85%	Relevan
Indomilk	63.26%	Relevan
Nestle Bear Brand	58.38%	Relevan
OT Fullo Grande (Assorted Wafer Sticks)	57.12%	Tidak

Tabel 6.11: Produk terkait ”Energen rasa kacang hijau” berdasarkan Euclidean Distance

Nama produk	Skor	Relevansi
Energen rasa vanila	73.25%	Relevan
Energen Susu Cereal Rasa Jahe	70.10%	Relevan
OT Fullo Grande (Assorted Wafer Sticks)	64.46%	Tidak
Indomie Mi Instant Rasa Soto Mie	64.17%	Tidak
Indomie Mi Rasa Ayam Bawang	63.71%	Tidak

Tabel 6.12: Produk terkait ”Good Day Original” berdasarkan Euclidean Distance

Nama produk	Skor	Relevansi
Good Day White Cappuccino	61.41%	Relevan
Good Day Tiramisu Bliss Coffee	60.93%	Relevan
Good Day Mocca Latte Coffee	58.96%	Relevan
Indomie Mi Instant Rasa Soto Mie	58.86%	Tidak
Good Day Coffee Drink Avocado	58.64%	Relevan

Tabel 6.13: Produk terkait ”Indomie Mi Instan Goreng” berdasarkan Euclidean Distance

Nama produk	Skor	Relevansi
Indomie Mi Goreng Rasa Sate	82.50%	Relevan
Indomie Mi Goreng Rendang	78.70%	Relevan
Indomie Mi Rasa Kari Ayam	78.69%	Relevan
Indomie Mi Rasa Ayam Special	78.69%	Relevan
Indomie Mi Rasa Ayam Bawang	78.46%	Relevan

Tabel 6.14: Produk terkait ”Majorico Wafer Roll” berdasarkan Euclidean Distance

Nama produk	Skor	Relevansi
Majorico Wafer Roll Rasa Pisang Coklat	73.60%	Relevan
Astor Wafer Stick Cokelat	70.37%	Relevan
Tim Tam Chocolate	67.59%	Tidak
OT Fullo Grande (Assorted Wafer Sticks)	67.56%	Relevan
Walls Cornetto Black and White	67.16%	Tidak

3. Pengujian relevansi MoreLikeThis (MLT)

Tabel 6.15: Produk terkait ”Ultra Milk Plain” berdasarkan MoreLikeThis

Nama produk	Skor	Relevansi
Ultra Milk Rasa Stroberi	30.296375	Relevan
Ultra Milk Minuman Susu UHT Rasa Coklat	25.9232	Relevan
Indomilk	19.19752	Relevan
Nestle Bear Brand	13.153961	Relevan
Frisian Flag Cokelat	12.609884	Relevan

Tabel 6.16: Produk terkait ”Energen rasa kacang hijau” berdasarkan MoreLikeThis

Nama produk	Skor	Relevansi

Energen rasa vanila	51.468487	Relevan
Energen Susu Cereal Rasa Jahe	45.12898	Relevan
Simba Corn Flakes	21.084887	Tidak
Dua Kelinci Hot Nut Corn Flavor Kacang Hawaii	20.136385	Tidak
OT Fullo Grande Assorted Wafer Sticks	19.523043	Tidak

Tabel 6.17: Produk terkait "Good Day Original" berdasarkan MoreLikeThis

Nama produk	Skor	Relevansi
Good Day Tiramisu Bliss Coffee	33.60615	Relevan
Nata De Coco	30.755249	Tidak
Good Day White Cappuccino	30.715242	Relevan
Luwak White Koffie	23.86652	Relevan
Good Day Coffee Drink Avocado	22.802986	Relevan

Tabel 6.18: Produk terkait "Indomie Mi Instan Goreng" berdasarkan MoreLikeThis

Nama produk	Skor	Relevansi
Indomie Mi Goreng Rasa Sate	88.16152	Relevan
Indomie Mi Goreng Rendang	70.11978	Relevan
Indomie Mi Rasa Ayam Special	59.74435	Relevan
Indomie Mi Rasa Kari Ayam	59.74435	Relevan
Indomie Mi Rasa Ayam Bawang	59.181973	Relevan

Tabel 6.19: Produk terkait "Majorico Wafer Roll" berdasarkan MoreLikeThis

Nama produk	Skor	Relevansi
Majorico Wafer Roll Rasa Pisang Coklat	72.69476	Relevan
Astor Wafer Stick Cokelat	37.12549	Relevan
Majorico Wafer Roll Rasa Susu Vanilla	35.18575	Relevan
OT Fullo Grande Assorted Wafer Sticks	33.156155	Relevan
Walls Cornetto Black and White	30.844032	Tidak

Secara keseluruhan, hasil pengujian akurasi dari algoritma Cosine, Euclidean, dan MoreLikeThis dapat dilihat pada tabel 6.20

Tabel 6.20: Hasil pengujian menggunakan Cosine, Euclidean, dan MoreLikeThis (MLT)

Nama produk	Cosine	Euclidean	MLT
Ultra Milk Plain	100%	80%	100%
Energen rasa kacang hijau	40%	40%	40%
Good Day Original	100%	80%	80%
Indomie Mi Instan Goreng	100%	100%	100%
Majorico Wafer Roll	80%	60%	80%
Rata-rata presisi	**84%**	72%	80%

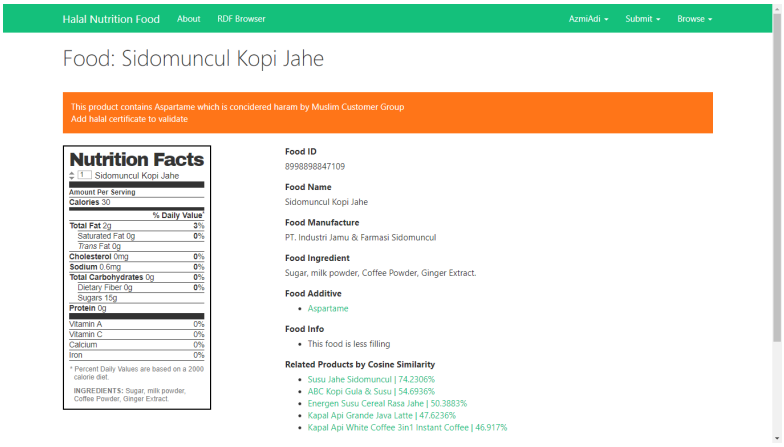
Pada tabel tersebut dapat dilihat bahwa pencarian produk terkait menggunakan euclidean distance memiliki rata-rata presisi sebesar 72%. Untuk pengujian menggunakan cosine similarity menghasilkan rata-rata presisi paling tinggi, yaitu sebesar 84%. Untuk pengujian menggunakan fitur MoreLikeThis didapatkan rata-rata presisi sebesar 80%.

6.1.3 Pengujian fungsional aplikasi

Pengujian fungsional aplikasi dilakukan dengan menggunakan skenario dalam penggunaan aplikasi Halal Nutrition Food untuk mengetahui apakah aplikasi dapat memberikan pesan notifikasi mengenai produk terkait sesuai yang diharapkan. Berikut adalah skenario pengujian yang dilakukan :

1. Notifikasi pada produk yang tidak memiliki sertifikat dan mengandung zat aditif dalam kategori haram. Pada pengujian ini diambil contoh foodproductId : 49. Berikut pada gambar 6.2

merupakan notifikasi yang ditampilkan pada halaman detil produk dengan foodproductId 49 :

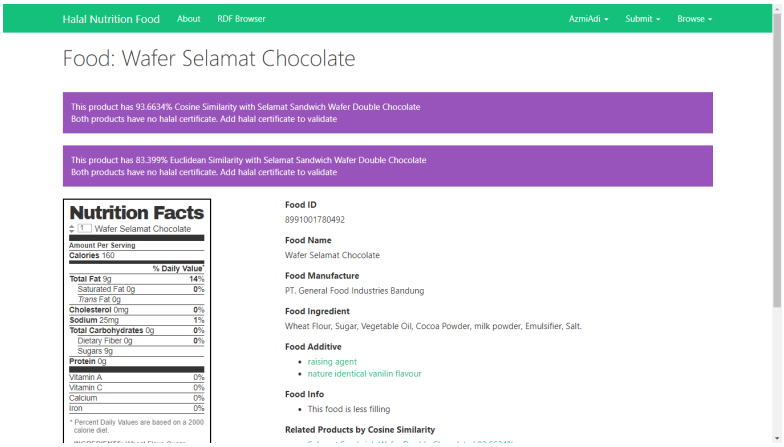


Gambar 6.2: Pengujian notifikasi pada produk yang tidak memiliki sertifikat dan mengandung zat aditif dalam kategori haram

Data produk tersebut menujukkan tidak ada sertifikat yang dimasukkan saat entri data. Data produk tersebut juga mengandung zat aditif dalam kategori haram (Aspartam). Aplikasi Halal Nutrition Food akan menampilkan notifikasi yang berisikan pesan bahwa produk yang sedang dilihat mengandung zat aditif dalam kategori haram menurut organisasi yang menyatakan bahwa zat aditif tersebut haram. Pada skenario ini produk dengan foodproductId 49 mengandung zat aditif yang diharamkan oleh Muslim Consumer Group.

2. Notifikasi pada produk yang tidak memiliki sertifikat, tidak mengandung zat aditif dalam kategori haram, memiliki kemiripan lebih dari 75% berdasarkan euclidean distance atau cosine similarity. Pada pengujian ini diambil contoh foodproductId : 20. Berikut pada gambar 6.3 merupakan notifikasi yang ditampilkan pada halaman detil produk dengan foodp-

roductId 20 :



Gambar 6.3: Pengujian notifikasi pada produk yang tidak memiliki sertifikat, tidak mengandung zat aditif dalam kategori haram, memiliki kemiripan lebih dari 75% berdasarkan euclidean distance atau cosine similarity

Data produk tersebut menujukkan terdapat produk terkait dengan kemiripan 93.6634% berdasarkan Cosine Similarity. Data produk tersebut juga menujukkan terdapat produk terkait dengan kemiripan 83.399% berdasarkan Euclidean Distance. Aplikasi Halal Nutrition Food akan menampilkan notifikasi yang berisikan pesan bahwa produk yang sedang dilihat memiliki kemiripan lebih besar dari 75% dengan produk lain berdasarkan Euclidean Distance, Cosine Similarity, maupun keduanya. Pada skenario ini produk dengan foodproductId 20 memiliki kemiripan lebih besar 75% dengan produk lain berdasarkan Euclidean Distance dan Cosine Similarity

6.2 Pembahasan

Pada subbab ini akan dibahas dan disimpulkan hasil dari pengujian fungsional dari perangkat lunak.

6.2.1 Pembahasan pengujian algoritma

Pada pengujian algoritma dapat dilihat bahwa nilai similarity yang dihitung dari nilai frekuensi term yang disimpan dalam index telah sesuai dengan rumus perhitungan euclidean distance dan cosine similarity.

6.2.2 Pembahasan pengujian relevansi

Pada pengujian euclidean distance, didapatkan nilai presisi yang kecil. Euclidean distance pada dasarnya juga mengukur jarak antara 2 titik pada ruang euclidean, yang dipengaruhi oleh bobot dari term. Konversi dari jarak ke ukuran similarity menghasilkan presisi lebih kecil dibanding cosine similarity.

Hal ini dikarenakan cosine similarity mengukur similarity berdasarkan sudut dari vektor yang dibentuk dari term dokumen. Perbandingan tidak dilakukan dengan membandingkan besarnya dua vektor yang merupakan representasi dari term, karena dua term yang identik dapat dianggap berbeda karena salah satu vektor lebih panjang dari yang lain.

Pengujian MoreLikeThis menghasilkan presisi yang cukup baik, namun masih lebih kecil dibandingkan dengan Cosine Similarity. MoreLikeThis merupakan fitur dari Apache Lucene, yang merupakan query builder yang dapat mencari dokumen yang serupa de-

ngan fitur yang diatur pada MoreLikeThis. Pada pengujian ini MoreLikeThis diatur agar mengambil field ”label” dan ”containsIngredient” dari dokumen, lalu menjadikannya sebagai term dalam query dan mencarinya ke dalam index. Dokumen yang memiliki term akan dibobotkan menggunakan BM25F dan diambil 5 dokumen teratas. Untuk mengambil 5 dokumen teratas, digunakan batasan 6, karena dokumen yang teratas dari query adalah dokumen yang ingin dicari itu sendiri. MoreLikeThis bukan merupakan algoritma untuk menghitung similarity, melainkan query builder yang dapat mengambil fitur dari dokumen yang ingin dicari.

6.2.3 Pembahasan pengujian fungsional aplikasi

Pada pengujian fungsional dapat dilihat bahwa aplikasi dapat menampilkan notifikasi dari skenario sesuai dengan yang diharapkan. Aplikasi dapat menampilkan notifikasi pada produk yang tidak memiliki sertifikat dan mengandung zat aditif dalam kategori haram. Aplikasi juga dapat menampilkan notifikasi pada produk yang tidak memiliki sertifikat maupun zat aditif haram, namun memiliki kemiripan lebih besar dari 75% dengan produk lain berdasarkan Cosine Similarity maupun Euclidean Distance.

BAB 7

KESIMPULAN DAN SARAN

Pada bab ini akan dijelaskan kesimpulan dan saran dalam pengerjaan tugas akhir.

7.1 Kesimpulan

Berdasarkan dengan pengerjaan tugas akhir dengan judul ”Pengembangan Pencarian Produk Terkait menggunakan Euclidean Distance dan Cosine Similarity pada Aplikasi Halal Nutrition Food” yang telah dilakukan dapat disimpulkan beberapa hal sebagai berikut:

1. Fitur pencarian produk terkait pada aplikasi Halal Nutrition Food berhasil dibuat. Fitur pencarian produk terkait menampilkan 5 produk yang pada halaman detil produk yang memiliki keterkaitan dari segi komposisi dan nama.
2. Daftar produk terkait berhasil dibuat. Daftar produk terkait dari seluruh produk dalam aplikasi Halal Nutrition Food dihasilkan dan disimpan dalam database MySQL untuk selanjutnya digunakan dalam query agar dapat ditampilkan pada halaman detil.
3. Cosine similarity dapat digunakan untuk menghasilkan daftar produk terkait dengan presisi paling tinggi dibandingkan Euclidean Distance dan MoreLikeThis dari Apache Lucene.
4. Aplikasi dapat menampilkan notifikasi apabila terdapat produk yang tidak memiliki sertifikat dan mengandung zat aditif haram, juga menampilkan notifikasi apabila terdapat produk yang tidak memiliki sertifikat maupun zat aditif haram, namun memiliki kemiripan dengan produk lain berdasarkan

Cosine Similarity maupun Euclidean Distance.

7.2 Saran

Saran penulis untuk penelitian selanjutnya sebagai berikut:

1. Melakukan pre-processing sebelum melakukan pembobotan term dari dokumen menggunakan algoritma string similarity, seperti Levenshtein, Jaccard, dan sebagainya untuk mengantisipasi apabila terjadi salah ketik.
2. Menyimpan komposisi makanan yang dikategorikan haram, seperti daging babi. Untuk saat ini aplikasi Halal Nutrition Food hanya menyimpan E-number dari zat aditif.

DAFTAR PUSTAKA

[1] Linkeddata. dapat diakses di https://www.w3.org/wiki/LinkedData.

[2] Renaud Delbru. Searching web data: an entity retrieval model.

[3] Adnan Mauludin Fajriyadi. Rancang bangun sistem pencarian dalam aplikasi halal nutrition food menggunakan algoritma okapi bm25f.

[4] Jauhar Fatawi. Rancang bangun perangkat lunak linked open data halal dan gizi pada produk makanan dan minuman.

[5] Bening et al Herwijayanti. Klasifikasi berita online dengan menggunakan pembobotan tf-idf dan cosine similarity.

[6] Kelvin. Basic concepts lucene, 2017.

[7] Giridhar Kumaran and James Allan. Text classification and named entities for new event detection. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, pages 297–304, New York, NY, USA, 2004. ACM.

[8] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[9] LPPOM MUI. Statistik sertifikasi halal indonesia, 2015.

[10] Republika. Produk pangan bersertifikat halal masih minim, 2017.

[11] Amol Sonawane. Using apache lucene to search text, 2009.

[12] Badan Pusat Statistik. Hasil sensus penduduk 2010, 2010.

[13] Deyi Xiong. Semantic similarity from natural language and ontology analysis. *Computational Linguistics*, 42(4):829–831, 2016.

BIODATA PENULIS

Penulis lahir di Surabaya pada tanggal 30 Mei 1996. Merupakan anak kedua dari 2 bersaudara dan telah menempuh pendidikan formal yaitu; SD Negeri Medokan Ayu 1 Surabaya, SMP Negeri 35 Surabaya, dan SMA Negeri 16 Surabaya.

Pada tahun 2014 melanjutkan pendidikan di Departemen Sistem Informasi FTIF - Institut Teknologi Sepuluh Nopember (ITS) Surabaya dan terdaftar sebagai mahasiswa dengan NRP 5214100045. Selama menjadi mahasiswa penulis mengikuti kegiatan kemahasiswaan seperti beberapa kepanitiaan ditingkat ITS dan Nasional serta aktif sebagai staff Departemen Aplikasi dan Teknologi HMSI FTIf periode 2015/2016. Disamping aktif dalam kegiatan kemahasiswaan, penulis juga pernah menjadi asisten praktikum pada mata kuliah bahasa pemrograman. Pada tahun keempat karena penulis tertarik dengan bidang desiminasi informasi, maka penulis mengambil bidang minat Laboratorium Akuisisi Data dan Diseminasi Informasi (ADDI). Penulis dapat dihubungi melalui email azmiadifirmansyah@gmail.com.